Latent Variable Modelling with Hyperbolic Normalizing Flows

Avishek Joey Bose [1 2] **Ariella Smofsky** [1 2] **Renjie Liao** [3 4] **Prakash Panangaden** [1 2] **William L. Hamilton** [1 2]

Abstract

The choice of approximate posterior distributions plays a central role in stochastic variational inference (SVI). One effective solution is the use of normalizing flows to construct flexible posterior distributions. However, a key limitation of existing normalizing flows is that they are restricted to Euclidean space and are ill-equipped to model data with an underlying hierarchical structure. To address this fundamental limitation, we present the first extension of normalizing flows to hyperbolic spaces. We first elevate normalizing flows to hyperbolic spaces using coupling transforms defined on the tangent bundle, termed Tangent Coupling ($\mathcal{TC}$). We further introduce Wrapped Hyperboloid Coupling ($\mathcal{W}\mathbb{H}C$), a fully invertible and learnable transformation that explicitly utilizes the geometric structure of hyperbolic spaces, allowing for expressive posteriors while being efficient to sample from. We demonstrate the efficacy of our novel normalizing flow over hyperbolic VAEs and Euclidean normalizing flows. Our approach achieves improved performance on density estimation, as well as reconstruction of real-world graph data, which exhibit a hierarchical structure. Finally, we show that our approach can be used to power a generative model over hierarchical data using hyperbolic latent variables.

1. Introduction

Stochastic variational inference (SVI) methods provide an appealing way of scaling probabilistic modeling to large scale data. These methods transform the problem of computing an intractable posterior distribution to finding the best approximation within a class of tractable probability distributions (Hoffman et al., 2013). Using tractable classes of approximate distributions, *e.g.*, mean-field, and Bethe

[1]McGill University [2]Mila [3]University of Toronto [4]Vector Institute. Correspondence to: Joey Bose <joey.bose@mail.mcgill.ca>, Ariella Smofsky <ariella.smofsky@mail.mcgill.ca>.

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

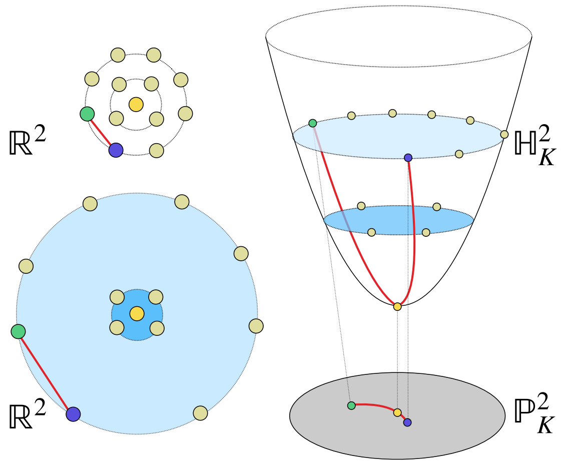

Figure 1. The shortest path between a given pair of node embeddings in $\mathbb{R}^2$ and hyperbolic space as modelled by the Lorentz model $\mathbb{H}^2_K$ and Poincaré disk $\mathbb{P}^2_K$. Unlike Euclidean space, distances between points grow exponentially as you move away from the origin in hyperbolic space, and thus the shortest paths between points in hyperbolic space go through a common parent node (i.e., the origin), giving rise to hierarchical and tree-like structure.

approximations, facilitates efficient inference, at the cost of limiting the expressiveness of the learned posterior.

In recent years, the power of these SVI methods has been further improved by employing *normalizing flows*, which greatly increase the flexibility of the approximate posterior distribution. Normalizing flows 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 (Rezende & Mohamed, 2015). Indeed, flow-based posteriors enjoy many advantages such as efficient sampling, exact likelihood estimation, and low-variance gradient estimates when the base distribution is reparametrizable, making them ideal for modern machine learning problems. There have been numerous advances in normalizing flow construction in Euclidean spaces, such as RealNVP (Dinh et al., 2017), B-NAF (Huang et al., 2018; De Cao et al., 2019), and FFJORD (Grathwohl et al., 2018), to name a few.

However, current normalizing flows are restricted to Euclidean space, and as a result, these approaches are ill-equipped to model data with an underlying hierarchical structure. Many real-world datasets—such as ontologies, social networks, sentences in natural language, and evolutionary relationships between biological entities in phylogenetics—exhibit rich hierarchical or tree-like structure. Hierarchical data of this kind can be naturally represented in hyperbolic spaces, *i.e.*, non-Euclidean spaces with constant negative curvature (Figure 1). But Euclidean normalizing flows fail to incorporate these structural inductive biases, since Euclidean space cannot embed deep hierarchies without suffering from high distortion (Sarkar, 2011). Furthermore, sampling from densities defined on Euclidean space will inevitability generate points that do not lie on the underlying hyperbolic space.

Present work. To address this fundamental limitation, we present the first extension of normalizing flows to hyperbolic spaces. Prior works have considered learning models with hyperbolic parameters (Liu et al., 2019b; Nickel & Kiela, 2018) as well as variational inference with hyperbolic latent variables (Nagano et al., 2019; Mathieu et al., 2019), but our work represents the first approach to allow flexible density estimation in hyperbolic space.

To define our normalizing flows we leverage the Lorentz model of hyperbolic geometry and introduce two new forms of coupling, *Tangent Coupling* ($\mathcal{TC}$) and *Wrapped Hyperboloid Coupling* ($\mathcal{W}\mathbb{H}\mathcal{C}$). These define flexible and invertible transformations capable of transforming sampled points in the hyperbolic space. We derive the change of volume associated with these transformations and show that it can be computed efficiently with $\mathcal{O}(n)$ cost, where n is the dimension of the hyperbolic space. We empirically validate our proposed normalizing flows on structured density estimation, reconstruction, and generation tasks on hierarchical data, highlighting the utility of our proposed approach.

2. Background on Hyperbolic Geometry

Within the Riemannian geometry framework, hyperbolic spaces are manifolds with constant negative curvature K and are of particular interest for embedding hierarchical structures. There are multiple models of n-dimensional hyperbolic space, such as the hyperboloid $\mathbb{H}_K^n$, also known as the Lorentz model, or the Poincaré ball $\mathbb{P}_K^n$. Figure 1 illustrates some key properties of $\mathbb{H}_K^2$ and $\mathbb{P}_K^2$, highlighting how distances grow exponentially as you move away from the origin and how the shortest paths between distant points tend to go through a common parent (*i.e.*, the origin), giving rise to a hierarchical or tree-like structure. In the next section, we briefly review the Lorentz model of hyperbolic geometry. We are not assuming a background in Riemannian geometry, though Appendix A and Ratcliffe (1994) are of use to the interested reader. Henceforth, for notational clarity, we use boldface font to denote points on the hyperboloid manifold.

2.1. Lorentz Model of Hyperbolic Geometry

An n-dimensional hyperbolic space, $\mathbb{H}_K^n$, is the unique, complete, simply-connected n-dimensional Riemannian manifold of constant negative curvature, K. For our purposes, the Lorentz model is the most convenient representation of hyperbolic space, since it is equipped with relatively simple explicit formulas and useful numerical stability properties (Nickel & Kiela, 2018). We choose the 2D Poincaré disk $\mathbb{P}_1^2$ to visualize hyperbolic space because of its conformal mapping to the unit disk. The Lorentz model embeds hyperbolic space $\mathbb{H}_K^n$ within the $n+1$-dimensional Minkowski space, defined as the manifold $\mathbb{R}^{n+1}$ equipped with the following inner product:

$$\langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{L}} := -x_0 y_0 + x_1 y_1 + \cdots + x_n y_n, \qquad (1)$$

which has the type $\langle \cdot, \cdot \rangle_{\mathcal{L}} : \mathbb{R}^{n+1} \times \mathbb{R}^{n+1} \to \mathbb{R}$. It is common to denote this space as $\mathbb{R}^{1,n}$ to emphasize the distinct role of the zeroth coordinate. In the Lorentz model, we model hyperbolic space as the (upper sheet of) the hyperboloid embedded in Minkowski space. It is a remarkable fact that though the Lorentzian metric (Eq. 1) is indefinite, the induced Riemannian metric $g_{\mathbf{x}}$ on the unit hyperboloid is positive definite (Ratcliffe, 1994). The n-Hyperbolic space with constant negative curvature K with origin $\mathbf{o} = (1/K, 0, \ldots, 0)$, is a Riemannian manifold $(\mathbb{H}_K^n, g_{\mathbf{x}})$ where

$$\mathbb{H}_K^n := \{x \in \mathbb{R}^{n+1} : \langle \mathbf{x}.\mathbf{x} \rangle_{\mathcal{L}} = 1/K,\ x_0 > 0,\ K < 0\}.$$

Equipped with this, the induced distance between two points $(\mathbf{x}, \mathbf{y})$ in $\mathbb{H}_K^n$ is given by

$$d(\mathbf{x}, \mathbf{y})_{\mathcal{L}} := \frac{1}{\sqrt{-K}} \operatorname{arccosh}(-K \langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{L}}). \qquad (2)$$

The tangent space to the hyperboloid at the point $\mathbf{p} \in \mathbb{H}_K^n$ can also be described as an embedded subspace of $\mathbb{R}^{1,n}$. It is given by the set of points satisfying the orthogonality relation with respect to the Minkowski inner product,[1]

$$\mathcal{T}_{\mathbf{p}} \mathbb{H}_K^n := \{u : \langle u, \mathbf{p} \rangle_{\mathcal{L}} = 0\} \qquad (3)$$

Of special interest are vectors in the tangent space at the origin of $\mathbb{H}_K^n$ whose norm under the Minkowski inner product is equivalent to the conventional Euclidean norm. That is $v \in \mathcal{T}_{\mathbf{o}} \mathbb{H}_K^n$ is a vector such that $v_0 = 0$ and $||\mathbf{v}||_{\mathcal{L}} := \sqrt{\langle \mathbf{v}, \mathbf{v} \rangle_{\mathcal{L}}} = ||\mathbf{v}||_2$. Thus *at the origin* the partial derivatives with respect to the ambient coordinates, $\mathbb{R}^{n+1}$, define the covariant derivative.

Projections. Starting from the extrinsic view by which we consider $\mathbb{R}^{n+1} \supset \mathbb{H}_K^n$, we may project any vector $x \in \mathbb{R}^{n+1}$

[1] It is also equivalently known as the Lorentz inner product.

on to the hyperboloid using the shortest Euclidean distance:

$$\text{proj}_{\mathbb{H}^n_K}(x) = \frac{x}{\sqrt{-K}||x||_{\mathcal{L}}}. \tag{4}$$

Furthermore, by definition a point on the hyperboloid satisfies $\langle \mathbf{x}, \mathbf{x} \rangle_{\mathcal{L}} = 1/K$ and thus when provided with n coordinates $\hat{x} = (x_1, \dots, x_n)$ we can always determine the missing coordinate to get a point on $\mathbb{H}^n_K$:

$$x_0 = \sqrt{||\hat{x}||_2^2 + \frac{1}{K}}. \tag{5}$$

Exponential Map. The exponential map takes a vector, v, in the tangent space of a point $\mathbf{x} \in \mathbb{H}^n_K$ to a point on the manifold—i.e., $\mathbf{y} = \exp^K_{\mathbf{x}}(v) : \mathcal{T}_{\mathbf{x}}\mathbb{H}^n_K \to \mathbb{H}^n_K$ by moving a unit length along the *geodesic*, γ (straightest parametric curve), uniquely defined by $\gamma(0) = \mathbf{x}$ with direction $\gamma'(0) = v$. The closed form expression for the exponential map is then given by

$$\exp^K_{\mathbf{x}}(v) = \cosh\left(\frac{||v||_{\mathcal{L}}}{R}\right)\mathbf{x} + \sinh\left(\frac{||v||_{\mathcal{L}}}{R}\right)\frac{Rv}{||v||_{\mathcal{L}}}, \tag{6}$$

where we used the *generalized radius* $R = 1/\sqrt{-K}$ in place of the curvature.

Logarithmic Map. As the inverse of the exponential map, the logarithmic map takes a point, $\mathbf{y}$, on the manifold back to the tangent space of another point $\mathbf{x}$ also on the manifold. In the Lorentz model this is defined as

$$\log^K_{\mathbf{x}} \mathbf{y} = \frac{\text{arccosh}(\alpha)}{\sqrt{\alpha^2 - 1}}(\mathbf{y} - \alpha\mathbf{x}), \tag{7}$$

where $\alpha = K\langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{L}}$.

Parallel Transport. The parallel transport for two points $\mathbf{x}, \mathbf{y} \in \mathbb{H}^n_K$ is a map that carries the vectors in $v \in \mathcal{T}_{\mathbf{x}}\mathbb{H}^n_K$ to corresponding vectors at $v' \in \mathcal{T}_{\mathbf{y}}\mathbb{H}^n_K$ along the geodesic. That is vectors are connected between the two tangent spaces such that the covariant derivative is unchanged. Parallel transport is a map that preserves the metric, *i.e.*, $\langle \text{PT}^K_{\mathbf{x}\to\mathbf{y}}(v), \text{PT}^K_{\mathbf{x}\to\mathbf{y}}(v') \rangle_{\mathcal{L}} = \langle v, v' \rangle_{\mathcal{L}}$ and in the Lorentz model is given by

$$\begin{aligned}\text{PT}^K_{\mathbf{x}\to\mathbf{y}}(v) &= v - \frac{\langle \log^K_{\mathbf{x}}(\mathbf{y}), v \rangle_{\mathcal{L}}}{d(\mathbf{x}, \mathbf{y})_{\mathcal{L}}}(\log^K_{\mathbf{x}}(\mathbf{y}) + \log^K_{\mathbf{y}}(\mathbf{x})) \\ &= v + \frac{\langle \mathbf{y}, v \rangle_{\mathcal{L}}}{R^2 - \langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{L}}}(\mathbf{x} + \mathbf{y}),\end{aligned} \tag{8}$$

where α is as defined above. Another useful property is that the inverse parallel transport simply carries the vectors back along the geodesic and is simply defined as $(\text{PT}^K_{\mathbf{x}\to\mathbf{y}}(v))^{-1} = \text{PT}^K_{\mathbf{y}\to\mathbf{x}}(v)$.

2.2. Probability Distributions on Hyperbolic Spaces

Probability distributions can be defined on Riemannian manifolds, which include $\mathbb{H}^n_K$ as a special case. One transforms the infinitesimal volume element on the manifold to the corresponding volume element in $\mathbb{R}^n$ as defined by the co-ordinate charts. In particular, given the Riemannian manifold $\mathcal{M}(\mathbf{z})$ and its metric $g_{\mathbf{z}}$, we have $\int p(\mathbf{z}) d\mathcal{M}(\mathbf{z}) = \int p(\mathbf{z})\sqrt{|g_{\mathbf{z}}|}d\mathbf{z}$, where $d\mathbf{z}$ is the Lebesgue measure. We now briefly survey three distinct generalizations of the normal distribution to Riemannian manifolds.

Riemannian Normal. The first is the Riemannian normal (Pennec, 2006; Said et al., 2014), which is derived from maximizing the entropy given a Fréchet mean μ and a dispersion parameter σ. Specifically, we have $\mathcal{N}_{\mathcal{M}}(\mathbf{z}|\mu, \sigma^2) = \frac{1}{Z}\exp\left(-d_{\mathcal{M}}(\mu, \mathbf{z})^2/2\sigma^2\right)$, where $d_{\mathcal{M}}$ is the *induced distance* and Z is the normalization constant (Said et al., 2014; Mathieu et al., 2019).

Restricted Normal. One can also restrict sampled points from the normal distribution in the ambient space to the manifold. One example is the Von Mises distribution on the unit circle and its generalized version, *i.e.*, Von Mises-Fisher distribution on the hypersphere (Davidson et al., 2018).

Wrapped Normal. Finally, we can define a wrapped normal distribution (Falorsi et al., 2019; Nagano et al., 2019), which is obtained by (1) sampling from $\mathcal{N}(0, I)$ and then transforming it to a point $v \in \mathcal{T}_{\mathbf{o}}\mathbb{H}^n_K$ by concatenating 0 as the zeroth coordinate; (2) parallel transporting the sample v from the tangent space at $\mathbf{o}$ to the tangent space of another point $\boldsymbol{\mu}$ on the manifold to obtain u; (3) mapping u from the tangent space to the manifold using the exponential map at $\boldsymbol{\mu}$. Sampling from such a distribution is straightforward and the probability density can be obtained via the change of variable formula,

$$\log p(\mathbf{z}) = \log p(v) - (n-1)\log\left(\frac{\sinh(\|u\|_{\mathcal{L}})}{\|u\|_{\mathcal{L}}}\right), \tag{9}$$

where $p(\mathbf{z})$ is the wrapped normal distribution and $p(v)$ is the normal distribution in the tangent space of $\mathbf{o}$.

3. Normalizing Flows on Hyperbolic Spaces

We seek to define flexible and learnable distributions on $\mathbb{H}^n_K$, which will allow us to learn rich approximate posterior distributions for hierarchical data. To do so, we design a class of invertible parametric hyperbolic functions, $f_i : \mathbb{H}^n_K \to \mathbb{H}^n_K$. A sample from the approximate posterior can then be obtained by first sampling from a simple base distribution $\mathbf{z}_0 \sim p(\mathbf{z})$ defined on $\mathbb{H}^n_K$ and then applying a composition of functions $f_{i\in[j]}$ from this class: $\mathbf{z}_j = f_j \circ f_{j-1} \circ \dots \circ f_1(\mathbf{z}_0)$.

In order to ensure effective and tractable learning, the class of functions f_i must satisfy three key desiderata:

1. Each function f_i must be invertible.
2. We must be able to efficiently sample from the final distribution, $\mathbf{z}_j = f_j \circ f_{j-1} \circ \cdots \circ f_1(\mathbf{z}_0)$.
3. We must be able to efficiently compute the associated change in volume (*i.e.*, the Jacobian determinant) of the overall transformation.

Given these requirements, the final transformed distribution is given by the change of variables formula:

$$\log p(\mathbf{z}_j) = \log p(\mathbf{z}_0) - \sum_{i=1}^{k} \log \det \left| \frac{\partial f_j}{\partial z_{j-1}} \right|. \tag{10}$$

Functions satisfying desiderata 1-3 in Euclidean space are often termed *normalizing flows* (Appendix B), and our work extends this idea to hyperbolic spaces. In the following sections, we describe two flows of increasing complexity: Tangent Coupling ($\mathcal{TC}$) and Wrapped Hyperboloid Coupling ($\mathcal{WHC}$). The first approach lifts a standard Euclidean flow to the tangent space at the origin of the hyperboloid. The second approach modifies the flow to explicitly utilize hyperbolic geometry. Figure 2 illustrates synthetic densities as learned by our approach on $\mathbb{P}_1^2$.

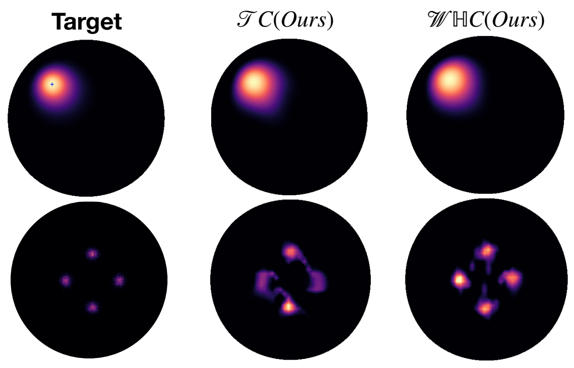

Figure 2. Comparison of density estimation in hyperbolic space for 2D wrapped Gaussian (WG) and mixture of wrapped gaussian (MWG) on $\mathbb{P}_1^2$. Densities are visualized in the Poincaré disk. Additional qualitative results can be found in Appendix F.

3.1. Tangent Coupling

Similar to the Wrapped Normal distribution (Section 2.2), one strategy to define a normalizing flow on the hyperboloid is to use the tangent space at the origin. That is, we first sample a point from our base distribution—which we define to be a Wrapped Normal—and use the logarithmic map at the origin to transport it to the corresponding tangent space. Once we arrive at the tangent space we are free to apply any Euclidean flow before finally projecting back to the manifold using the exponential map. This approach leverages the fact that the tangent bundle of a hyperbolic manifold has a well-defined vector space structure, allowing affine transformations and other operations that are ill-defined on the manifold itself.

Following this idea, we build upon one of the earliest and most well-studied flows: the RealNVP flow (Dinh et al., 2017). At its core, the RealNVP flow uses a computationally symmetric transformation (affine coupling layer), which has the benefit of being fast to evaluate and invert due to its lower triangular Jacobian, whose determinant is cheap to compute. Operationally, the coupling layer is implemented using a binary mask, and partitions some input $\tilde{x}$ into two sets, where the first set, $\tilde{x}_1 := \tilde{x}_{1:d}$, is transformed elementwise independently of other dimensions. The second set, $\tilde{x}_2 := \tilde{x}_{d+1:n}$, is also transformed elementwise but in a way that depends on the first set (see Appendix B.2 for more details). Since all coupling layer operations occur at $\mathcal{T}_{\mathbf{o}}\mathbb{H}_K^n$ we term this form of coupling as Tangent Coupling ($\mathcal{TC}$).

Thus, the overall transformation due to one layer of our $\mathcal{TC}$ flow is a composition of a logarithmic map, affine coupling defined on $\mathcal{T}_{\mathbf{o}}\mathbb{H}_k^n$, and an exponential map:

$$\tilde{f}^{\mathcal{TC}}(\tilde{x}) = \begin{cases} \tilde{z}_1 &= \tilde{x}_1 \\ \tilde{z}_2 &= \tilde{x}_2 \odot \sigma(s(\tilde{x}_1)) + t(\tilde{x}_1) \end{cases}$$
$$f^{\mathcal{TC}}(\mathbf{x}) = \exp_{\mathbf{o}}^K(\tilde{f}^{\mathcal{TC}}(\log_{\mathbf{o}}^K(\mathbf{x}))), \tag{11}$$

where $\tilde{x} = \log_{\mathbf{o}}^K(\mathbf{x})$ is a point on $\mathcal{T}_{\mathbf{o}}\mathbb{H}_K^n$, and σ is a pointwise non-linearity such as the exponential function. Functions s and t are parameterized scale and translation functions implemented as neural nets from $\mathcal{T}_{\mathbf{o}}\mathbb{H}_K^d \to \mathcal{T}_{\mathbf{o}}\mathbb{H}_K^{n-d}$. One important detail is that arbitrary operations on a tangent vector $v \in \mathcal{T}_{\mathbf{o}}\mathbb{H}_K^n$ may transport the resultant vector outside the tangent space, hampering subsequent operations. To avoid this we can keep the first dimension fixed at $v_0 = 0$ to ensure we remain in $\mathcal{T}_{\mathbf{o}}\mathbb{H}_K^n$.

Similar to the Euclidean RealNVP, we need an efficient expression for the Jacobian determinant of $f^{\mathcal{TC}}$.

Proposition 1. *The Jacobian determinant of a single $\mathcal{TC}$ layer in equation 11 is:*

$$\left| \det\left(\frac{\partial \mathbf{y}}{\partial \boldsymbol{x}} \right) \right| = \left(\frac{R \sinh(\frac{||z||_{\mathcal{L}}}{R})}{||z||_{\mathcal{L}}} \right)^{n-1} \times \prod_{i=d+1}^{n} \sigma(s(\tilde{x}_1))_i$$
$$\times \left(\frac{R \sinh(\frac{||\log_{\boldsymbol{o}}^K(\boldsymbol{x})||_{\mathcal{L}}}{R})}{||\log_{\boldsymbol{o}}^K(\boldsymbol{x})||_{\mathcal{L}}} \right)^{1-n} \tag{12}$$

where, $\mathbf{z} = \tilde{f}^{\mathcal{TC}}(\tilde{x})$ and $\tilde{f}^{\mathcal{TC}}$ is as defined above.

Proof Sketch. Here we only provide a sketch of the proof and details can be found in Appendix C. First, observe that the overall transformation is a valid composition of functions: $\mathbf{y} := \exp_{\mathbf{o}}^K \circ \tilde{f}^{\mathcal{TC}} \circ \log_{\mathbf{o}}^K(\mathbf{x})$. Thus, the overall determinant can be computed by chain rule and the identity, $\det\left(\frac{\partial \mathbf{y}}{\partial \mathbf{x}}\right) = \det\left(\frac{\partial \exp_{\mathbf{o}}^K(z)}{\partial z}\right) \cdot \det\left(\frac{\partial f(\tilde{x})}{\partial \tilde{x}}\right) \cdot \det\left(\frac{\partial \log_{\mathbf{o}}^K(\mathbf{x})}{\partial \mathbf{x}}\right)$. Tackling each function in the composition individually,

$\det\left(\frac{\partial \exp_{\mathbf{o}}^{K}(z)}{\partial z}\right) = \left(\frac{R\sinh(\frac{||z||_{\mathcal{L}}}{R})}{||z||_{\mathcal{L}}}\right)^{n-1}$ as derived in Skopek et al. (2019). As the logarithmic map is the inverse of the exponential map the Jacobian determinant is simply the inverse of the determinant of the exponential map, which gives the $\det\left(\frac{\partial \log_{\mathbf{o}}^{K}(\mathbf{x})}{\partial \mathbf{x}}\right)$ term. For the middle term, we must calculate the directional derivative of $\tilde{f}^{\mathcal{T}C}$ in an orthonormal basis w.r.t. the Lorentz inner product, of $\mathcal{T}_{\mathbf{o}}\mathbb{H}_K^n$. Since the standard Euclidean basis vectors $e_1, ..., e_n$ are also a basis for $\mathcal{T}_{\mathbf{o}}\mathbb{H}_K^n$, the Jacobian determinant $\det\left(\frac{\partial f(\tilde{x})}{\partial \tilde{x}}\right)$ simplifies to that of the RealNVP flow, which is lower triangluar and is thus efficiently computable in $\mathcal{O}(n)$ time.

□

It is remarkable that the middle term in Proposition 1 is precisely the same change in volume associated with affine coupling in RealNVP. The change in volume due to the hyperbolic space only manifests itself through the exponential and logarithmic maps, each of which can be computed in $\mathcal{O}(n)$ cost. Thus, the overall cost is only slightly larger than the regular Euclidean RealNVP, but still $\mathcal{O}(n)$.

3.2. Wrapped Hyperboloid Coupling

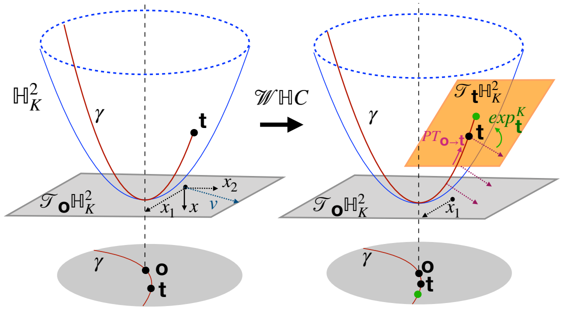
$\mathbb{H}_K^2$
γ
$\mathbf{t}$
$\mathcal{T}_{\mathbf{o}}\mathbb{H}_K^2$
x_1
x_2
x
v
$\mathbf{o}$
$\mathcal{W}\mathbb{H}C$
$\mathcal{T}_{\mathbf{t}}\mathbb{H}_K^2$
$exp_{\mathbf{t}}^K$
$PT_{\mathbf{o}\to\mathbf{t}}$

Figure 3. Wrapped Hyperbolic Coupling. The left figure depicts a partitioned input point $\tilde{x}_1 := \tilde{x}_{1:d}$ and $\tilde{x}_2 := \tilde{x}_{d+1:n}$ prior to parallel transport. The right figure depicts the $\tilde{x}_2$ vector after it is transformed, parallel transported, and projected to $\mathbb{H}_K^n$.

The hyperbolic normalizing flow with $\mathcal{T}C$ layers discussed above operates purely in the tangent space at the origin. This simplifies the computation of the Jacobian determinant, but anchoring the flow at the origin may hinder its expressive power and its ability to leverage disparate regions of the manifold. In this section, we remedy this shortcoming with a new hyperbolic flow that performs translations between tangent spaces via parallel transport.

We term this transformation *Wrapped Hyperboloid Coupling* ($\mathcal{W}\mathbb{H}C$). As with the $\mathcal{T}C$ layer, it is a fully invertible transformation $f^{\mathcal{W}\mathbb{H}C} : \mathbb{H}_k^n \to \mathbb{H}_k^n$ with a tractable analytic form for the Jacobian determinant. To define a $\mathcal{W}\mathbb{H}C$ layer we first use the logarithmic map at the origin to transport a point to the tangent space. We employ the coupling strategy previously discussed and partition our input vector into two components: $\tilde{x}_1 := \tilde{x}_{1:d}$ and $\tilde{x}_2 := \tilde{x}_{d+1:n}$. Let $\tilde{x} = \log_{\mathbf{o}}^K(\mathbf{x})$ be the point on $\mathcal{T}_{\mathbf{o}}\mathbb{H}_K^n$ after the logarithmic map. The remainder of the $\mathcal{W}\mathbb{H}C$ layer can be defined as follows;

$$\tilde{f}^{\mathcal{W}\mathbb{H}C}(\tilde{x}) = \begin{cases} \tilde{z}_1 &= \tilde{x}_1 \\ \tilde{z}_2 &= \log_{\mathbf{o}}^K\left(\exp_{t(\tilde{x}_1)}^K\left(\mathrm{PT}_{\mathbf{o}\to t(\tilde{x}_1)}(v)\right)\right) \end{cases}$$
$$v = \tilde{x}_2 \odot \sigma(s(\tilde{x}_1))$$
$$f^{\mathcal{W}\mathbb{H}C}(\mathbf{x}) = \exp_{\mathbf{o}}^K(\tilde{f}^{\mathcal{W}\mathbb{H}C}(\log_{\mathbf{o}}^K(\mathbf{x}))). \tag{13}$$

Functions $s : \mathcal{T}_{\mathbf{o}}\mathbb{H}_k^d \to \mathcal{T}_{\mathbf{o}}\mathbb{H}_k^{n-d}$ and $t : \mathcal{T}_{\mathbf{o}}\mathbb{H}_k^d \to \mathbb{H}_k^n$ are taken to be arbitrary neural nets, but the role of t when compared to $\mathcal{T}C$ is vastly different. In particular, the generalization of translation on Riemannian manifolds can be viewed as parallel transport to a different tangent space. Consequently, in Eq. 13, the function t predicts a point on the manifold that we wish to parallel transport to. This greatly increases the flexibility as we are no longer confined to the tangent space at the origin. The logarithmic map is then used to ensure that both $\tilde{z}_1$ and $\tilde{z}_2$ are in the same tangent space before the final exponential map that projects the point to the manifold.

One important consideration in the construction of t is that it should only parallel transport functions of $\tilde{x}_2$. However, the output of t is a point on $\mathbb{H}_k^n$ and without care this can involve elements in $\tilde{x}_1$. To prevent such a scenario we construct the output of $t = [t_0, 0, \ldots, 0, t_{d+1}, \ldots, t_n]$ where elements $t_{d+1:n}$ are used to determine the value of t_0 using Eq. 5, such that it is a point on the manifold and every remaining index is set to zero. Such a construction ensures that only components of any function of $\tilde{x}_2$ are parallel transported as desired. Figure 3 illustrates the transformation performed by the $\mathcal{W}\mathbb{H}C$ layer.

Inverse of $\mathcal{W}\mathbb{H}C$. To invert the flow it is sufficient to show that argument to the final exponential map at the origin itself is invertible. Furthermore, note that $\tilde{x}_1$ undergoes an identity mapping and is trivially invertible. Thus, we need to show that the second partition is invertible, *i.e.* that the following transformation is invertible:

$$\tilde{z}_2 = \log_{\mathbf{o}}^K\left(\exp_{t(\tilde{x}_1)}^K\left(\mathrm{PT}_{\mathbf{o}\to t(\tilde{x}_1)}(v)\right)\right). \tag{14}$$

As discussed in Section 2, the parallel transport, exponential map, and logarithmic map all have well-defined inverses with closed forms. Thus, the overall transformation is invertible in closed form:

$$\begin{cases} \tilde{x}_1 &= \tilde{z}_1 \\ \tilde{x}_2 &= \left(\mathrm{PT}_{t(\tilde{z}_1)\to\mathbf{o}}(\log_{t(\tilde{z}_1)}^K(\exp_{\mathbf{o}}^K(\tilde{z}_2))\right) \odot \sigma(s(\tilde{z}_1))^{-1} \end{cases}$$

Properties of $\mathcal{W}\mathbb{H}C$. To compute the Jacobian determinant of the full transformation in Eq. 13 we proceed by analyzing the effect of $\mathcal{W}\mathbb{H}C$ on valid orthonormal bases w.r.t. the Lorentz inner product for the tangent space at the origin. We state our main result here and provide a sketch of the proof, while the entire proof can be found in Appendix D.

Proposition 2. *The Jacobian determinant of the function $\tilde{f}^{\mathcal{W}\mathbb{H}C}$ in equation 13 is:*

$$\left|\det\left(\frac{\partial \mathbf{y}}{\partial \mathbf{x}}\right)\right| = \prod_{i=d+1}^{n} \sigma(s(\tilde{x}_1))_i \times \left(\frac{R\sinh(\frac{||q||_{\mathcal{L}}}{R})}{||q||_{\mathcal{L}}}\right)^{l} \times \left(\frac{R\sinh(\frac{||\log_{\boldsymbol{o}}^{K}(\hat{\boldsymbol{q}})||_{\mathcal{L}}}{R})}{||\log_{\boldsymbol{o}}^{K}(q)||_{\mathcal{L}}}\right)^{-l} \times \left(\frac{R\sinh(\frac{||\tilde{z}||_{\mathcal{L}}}{R})}{||\tilde{z}||_{\mathcal{L}}}\right)^{n-1} \times \left(\frac{R\sinh(\frac{||\log_{\boldsymbol{o}}^{K}(\boldsymbol{x})||_{\mathcal{L}}}{R})}{||\log_{\boldsymbol{o}}^{K}(\boldsymbol{x})||_{\mathcal{L}}}\right)^{1-n}, \quad (15)$$

where $\tilde{z} = \text{concat}(\tilde{z}_1, \tilde{z}_2)$, the constant $l = n - d$, σ is a non-linearity, $q = \text{PT}_{\boldsymbol{o}\to t(\tilde{x}_1)}(v)$ and $\hat{\boldsymbol{q}} = \exp_{\boldsymbol{t}}^{K}(q)$.

Proof Sketch. We first note that the exponential and logarithmic maps applied at the beginning and end of the $\mathcal{W}\mathbb{H}C$ can be dealt with by appealing to the chain rule and the known Jacobian determinants for these functions as used in Proposition 1. Thus, what remains is the following term: $\left|\det\left(\frac{\partial z}{\partial \tilde{x}}\right)\right|$. To evaluate this term we rely on the following Lemma.

Lemma 1. *Let $h : \mathcal{T}_{\mathbf{o}}\mathbb{H}_k^n \to \mathcal{T}_{\mathbf{o}}\mathbb{H}_k^n$ be a function defined as:*

$$h(\tilde{x}) = z = \text{concat}(\tilde{z}_1, \tilde{z}_2). \quad (16)$$

Now, define a function $h^* : \mathcal{T}_{\mathbf{o}}\mathbb{H}^{n-d} \to \mathcal{T}_{\mathbf{o}}\mathbb{H}^{n-d}$ which acts on the subspace of $\mathcal{T}_{\mathbf{o}}\mathbb{H}^{n-d}$ corresponding to the standard basis elements $e_{d+1}, ..., e_n$ as

$$h^*(\tilde{x}_2) = \log_{\mathbf{o}_2}^{K}\left(\exp_{\mathbf{t}_2}^{K}(\text{PT}_{\mathbf{o}_2\to\mathbf{t}_2}(v))\right), \quad (17)$$

where $\tilde{x}_2$ denotes the portion of the vector $\tilde{x}$ corresponding to the standard basis elements $e_{d+1}, ..., e_n$ and s and t are constants (which depend on $\tilde{x}_1$). In Equation equation 17, we use $\mathbf{o}_2 \in \mathbb{H}^{n-d}$ to denote the vector corresponding to only the dimensions $d+1, ..., n$ and similarly for $\mathbf{t}_2$. Then we have that

$$\left|\det\left(\frac{\partial z}{\partial \tilde{x}}\right)\right| = \left|\det\left(\frac{\partial h^*(\tilde{x}_{d+1:n})}{\partial \tilde{x}_{d+1:n})}\right)\right|. \quad (18)$$

The proof for Lemma 1 is provided in Appendix D. Using Lemma 1, and the fact that $|\det(\text{PT}_{\mathbf{u}\to\mathbf{t}}(v))| = 1$ (Nagano et al., 2019) we are left with another composition of functions but on the subspace $\mathcal{T}_{\mathbf{o}}\mathbb{H}^{n-d}$. The Jacobian determinant for these functions, are simply that of the logarithmic map, exponential map and the argument to the parallel transport which can be easily computed as $\prod_{i=d+1}^{n} \sigma(s(\tilde{x}_1))$. □

The cost of computing the change in volume for one $\mathcal{W}\mathbb{H}C$ layer is $\mathcal{O}(n)$ which is the same as a $\mathcal{T}C$ layer plus the added cost of the two new maps that operate on the lower subspace of basis elements.

4. Experiments

We evaluate our $\mathcal{T}C$-flow and $\mathcal{W}\mathbb{H}C$-flow on three tasks: structured density estimation, graph reconstruction, and graph generation.[2] Throughout our experiments, we rely on three main baselines. In Euclidean space, we use Gaussian latent variables and affine coupling flows (Dinh et al., 2017), denoted $\mathcal{N}$ and $\mathcal{N}C$, respectively. In the Lorentz model, we use Wrapped Normal latent variables, $\mathbb{H}$-VAE, as an analogous baseline (Nagano et al., 2019). Since all model parameters are defined on Euclidean tangent spaces, models can be trained with conventional optimizers like Adam (Kingma & Ba, 2014). Following previous work, we also consider the curvature K as a learnable parameter with a warmup of 10 epochs, and we clamp the max norm of vectors to 40 before any logarithmic or exponential map (Skopek et al., 2019). Appendix E contains details on model architectures and implementation details.

4.1. Structured Density Estimation

We first consider structured density estimation in a canonical VAE setting (Kingma & Welling, 2013), where we seek to learn rich approximate posteriors using normalizing flows and evaluate the marginal log-likelihood of test data. Following work on hyperbolic VAEs, we test the approaches on a branching diffusion process (BDP) and dynamically binarized MNIST (Mathieu et al., 2019; Skopek et al., 2019).

To estimate the log likelihood we perform importance sampling using 500 samples from the test set (Burda et al., 2015). Our results are shown in Tables 1 and 2. On both datasets we observe our hyperbolic flows provide improvements when using latent spaces of low dimension. This result matches theoretical expectations—*e.g.*, that trees can be perfectly embedded in $\mathbb{H}_K^2$—and dovetails with previous work on graph embedding (Nickel & Kiela, 2017), thus highlighting the benefit of leveraging hyperbolic space is most prominent in small dimensions. However, as we increase the latent dimension, the Euclidean approaches can compensate for this intrinsic geometric limitation. In the case of BDP we note that the data is indeed a noisy binary tree, which theoretically can be represented in a 2-D hyperbolic space and thus moving to higher dimensional latent space is not beneficial.

[2] https://github.com/joeybose/HyperbolicNF

Model	BDP-2	BDP-4	BDP-6
$\mathcal{N}$-VAE	$-55.4_{\pm 0.2}$	$-55.2_{\pm 0.3}$	$-56.1_{\pm 0.2}$
$\mathbb{H}$-VAE	$\mathbf{-54.9}_{\pm 0.3}$	$-55.4_{\pm 0.2}$	$-58.0_{\pm 0.2}$
$\mathcal{NC}$	$-55.4_{\pm 0.4}$	**-54.7**$_{\pm 0.1}$	**-55.2**$_{\pm 0.3}$
$\mathcal{TC}$	**-54.9**$_{\pm 0.1}$	$-55.4_{\pm 0.1}$	$-57.5_{\pm 0.2}$
$\mathcal{W}\mathbb{H}\mathcal{C}$	**-55.1**$_{\pm 0.4}$	$-55.2_{\pm 0.2}$	$-56.9_{\pm 0.4}$

Table 1. Test Log Likelihood on Binary Diffusion Process versus latent dimension. All normalizing flows use 2-coupling layers.

Model	MNIST 2	MNIST 4	MNIST 6
$\mathcal{N}$-VAE	$-139.5_{\pm 1.0}$	$-115.6_{\pm 0.2}$	$-100.0_{\pm 0.02}$
$\mathbb{H}$-VAE	$*$	$-113.7_{\pm 0.9}$	$-99.8_{\pm 0.2}$
$\mathcal{NC}$	$-139.2_{\pm 0.4}$	$-115.2_{\pm 0.6}$	**-98.7**$_{0.3}$
$\mathcal{TC}$	$*$	**-112.5**$_{\pm 0.2}$	$-99.3_{\pm 0.2}$
$\mathcal{W}\mathbb{H}\mathcal{C}$	**-136.5**$_{\pm 2.1}$	**-112.8**$_{\pm 0.5}$	$-99.4_{\pm 0.2}$

Table 2. Test Log Likelihood on MNIST averaged over 5 runs verus latent dimension. * indicates numerically unstable settings.

4.2. Graph Reconstruction

We evaluate the utility of our hyperbolic flows by conducting experiments on the task of link prediction using graph neural networks (GNNs) (Scarselli et al., 2008) as an inference model. Given a simple graph $\mathcal{G} = (\mathcal{V}, A, X)$, defined by a set of nodes $\mathcal{V}$, an adjacency matrix $A \in \mathbb{Z}^{|\mathcal{V}|\times|\mathcal{V}|}$ and node feature matrix $X \in \mathbb{R}^{|\mathcal{V}|\times n}$, we learn a VGAE (Kipf & Welling, 2016) model whose inference network, q_ϕ, defines a distribution over node embeddings $q_\phi(Z|A, X)$. To score the likelihood of an edge existing between pairs of nodes we use an inner product decoder: $p(A_{u,v} = 1|z_u, z_v) = \sigma(z_u^T z_v)$, with dot products computed in $\mathcal{T}_{\mathbf{o}}\mathbb{H}^n_K$ when necessary. Given these components, the inference GNNs are trained to maximize the variational lower bound on a training set of edges.

We use two different disease datasets taken from (Chami et al., 2019) and (Mathieu et al., 2019)[3] for evaluation purposes. Our chosen datasets reflect important real world use cases where the data is known to contain hierarchies. One such measure to determine how tree-like a given graph is known to be Gromov's δ-hyperbolicity and traditional link prediction datasets such as Cora and Pubmed (Yang et al., 2016) were found to lack such a property and are not suitable candidates to evaluate our proposed approach (Chami et al., 2019). The first dataset Diseases-I is composed of a network of disorders and disease genes linked by the known disorder–gene associations (Goh et al., 2007). In the second dataset Diseases-II, we build tree networks of a SIR disease spreading model (Anderson et al., 1992), where node features determine the susceptibility to the disease. In Table 3 we report the AUC and average precision (AP) on the test set. We observe consistent improvements when using hyperbolic $\mathcal{W}\mathbb{H}\mathcal{C}$ flow. Similar to the structured density estimation setting, the performance gains of $\mathcal{W}\mathbb{H}\mathcal{C}$ are best observed in low-dimensional latent spaces.

[3]We uncovered issues with the two remaining datasets in (Mathieu et al., 2019) and thus omit them (Appendix G).

Model	Dis-I AUC	Dis-I AP	Dis-II AUC	Dis-II AP
$\mathcal{N}$-VAE	$0.90_{\pm 0.01}$	$0.92_{\pm 0.01}$	$0.92_{\pm 0.01}$	$0.91_{\pm 0.01}$
$\mathbb{H}$-VAE	$0.91_{\pm 5e\text{-}3}$	$0.92_{\pm 5e\text{-}3}$	$0.92_{\pm 4e\text{-}3}$	$0.91_{\pm 0.01}$
$\mathcal{NC}$	$0.92_{\pm 0.01}$	$0.93_{\pm 0.01}$	$0.95_{\pm 4e\text{-}3}$	$0.93_{\pm 0.01}$
$\mathcal{TC}$	**0.93**$_{\pm 0.01}$	$0.93_{\pm 0.01}$	**0.96**$_{\pm 0.01}$	$0.95_{\pm 0.01}$
$\mathcal{W}\mathbb{H}\mathcal{C}$	**0.93**$_{\pm 0.01}$	**0.94**$_{\pm 0.01}$	**0.96**$_{\pm 0.01}$	**0.96**$_{\pm 0.01}$

Table 3. Test AUC and Test AP on Graph Embeddings where Dis-I has latent dimesion 6 and Dis-II has latent dimension 2.

4.3. Graph Generation

Finally, we explore the utility of our hyperbolic flows for generating hierarchical structures. As a synthetic testbed, we construct datasets containing uniformly random trees as well as uniformly random lobster graphs (Golomb, 1996), where each graph contains between 20 to 100 nodes. Unlike prior work on graph generation—*i.e.,* (Liu et al., 2019a)—our datasets are designed to have explicit hierarchies, thus enabling us to test the utility of hyperbolic generative models. We then train a generative model to learn the distribution of these graphs. We expect the hyperbolic flows to provide a significant benefit for generating valid random trees, as well as learning the distribution of lobster graphs, which are a special subset of trees.

We follow the two-stage training procedure outlined in Graph Normalizing Flows (Liu et al., 2019a) in that we first train an autoencoder to give node-level latents on which we train an normalizing flow for density estimation. Empirically, we find that using GRevNets (Liu et al., 2019a) and defining edge probabilities using a distance-based decoder consistently leads to better generation performance. Thus, we define edge probabilities as $p(A_{u,v} = 1|z_u, z_v) = \sigma((-d_{\mathcal{G}}(u, v) - b)/\tau)$ where b and τ are learned edge specific bias and temperature parameters. At inference time, we first sample the number of nodes to generate from the empirical distribution of the dataset. We then independently sample node latents from our prior, beginning with a fully connected graph, and then push these samples through our learned flow to give refined edge probabilities.

To evaluate the various approaches, we construct 100 training graphs for each dataset to train our model. Figure 4 shows representative samples generated by the various approaches. We see that hyperbolic normalizing flows learn to generate tree-like graphs and also match the specific properties of the lobster graph distribution, whereas the Euclidean flow model tends to generate densely connected graphs with many cycles (or else disconnected graphs). To quantify these intuitions, Table 4 contains statistics on how often

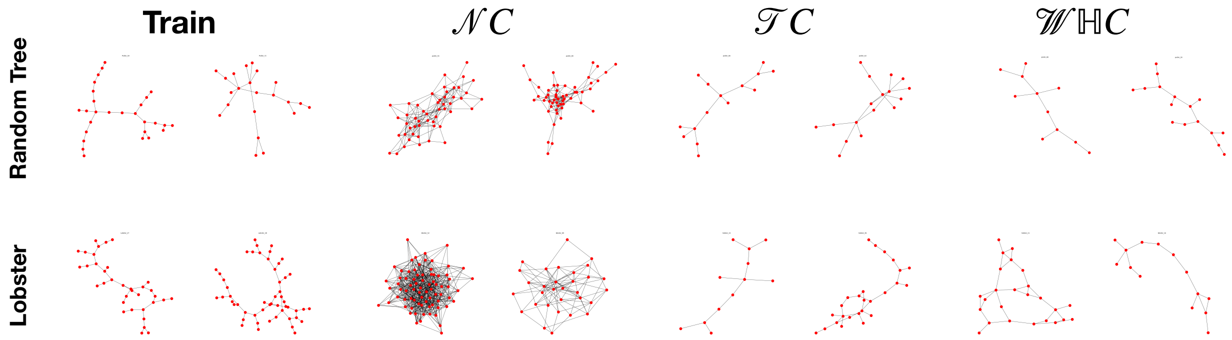

Figure 4. Selected qualitative results on graph generation for lobster and random tree graph.

Model	Accuracy	Avg. Clust.	Avg. GC.
$\mathcal{N}C$	$56.6_{\pm 5.5}$	$40.9_{\pm 42.7}$	$0.34_{\pm 0.10}$
$\mathcal{T}C$	$32.1_{\pm 1.9}$	$98.3_{\pm 89.5}$	$0.25_{\pm 0.12}$
$\mathcal{W}\mathbb{H}C$	$\mathbf{62.1}_{\pm 10.9}$	$\mathbf{21.1}_{\pm 13.4}$	$\mathbf{0.13}_{\pm 0.07}$

Table 4. Generation statistics on random trees over 5 runs.

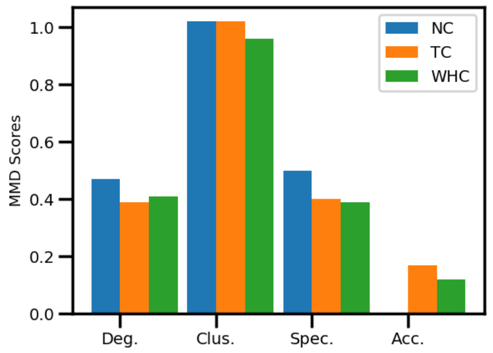

Figure 5. MMD scores for graph generation on Lobster graphs. Note, that $\mathcal{N}C$ achieves 0% accuracy.

the different models generate valid trees (denoted by "accuracy"), as well as the average number of triangles and the average global clustering coefficients for the generated graphs. Since the target data is random trees, a perfect model would achieve 100% accuracy, with no triangles, and a global clustering of 0 for all graphs. As a representative Euclidean baseline we employ Graph Normalizing Flows (GNFs) which is denoted as $\mathcal{N}C$ in Table 4 and Figure 5. We see that the hyperbolic models generate valid trees more often, and they generate graphs with fewer triangles and lower clustering on average. Finally, to evaluate how well the models match the specific properties of the lobster graphs, we follow Liao et al. (2019) and report the MMD distance between the generated graphs and a test set for various graph statistics (Figure 5). Again, we see that the hyperbolic approaches significantly outperform the Euclidean normalizing flow.

5. Related Work

Hyperbolic Geometry in Machine Learning:. The intersection of hyperbolic geometry and machine learning has recently risen to prominence (Dhingra et al., 2018; Tay et al., 2018; Law et al., 2019; Khrulkov et al., 2019; Ovinnikov, 2019). Early prior work proposed to embed data into the Poincaré ball model (Nickel & Kiela, 2017; Chamberlain et al., 2017). The equivalent Lorentz model was later shown to have better numerical stability properties (Nickel & Kiela, 2018), and recent work has leveraged even more stable tiling approaches (Yu & De Sa, 2019). In addition, there exists a burgeoning literature of hyperbolic counterparts to conventional deep learning modules on Euclidean spaces (*e.g.*, matrix multiplication), enabling the construction of hyperbolic neural networks (HNNs) (Gulcehre et al., 2018; Ganea et al., 2018) with further extensions to graph data using hyperbolic GNN architectures (Liu et al., 2019a; Chami et al., 2019). Latent variable models on hyperbolic space have also been investigated in the context of VAEs, using generalizations of the normal distribution (Nagano et al., 2019; Mathieu et al., 2019). In contrast, our work learns a flexible approximate posterior using a novel normalizing flow designed to use the geometric structure of hyperbolic spaces. In addition to work on hyperbolic VAEs, there are also several works that explore other non-Euclidean spaces (e.g., spherical VAEs) (Davidson et al., 2018; Falorsi et al., 2019; Grattarola et al., 2019).

Learning Implicit Distributions. In contrast with exact likelihood methods there is growing interest in learning implicit distributions for generative modelling. Popular approaches include density ratio estimation methods using a parametric classifiers such as GANS (Goodfellow et al., 2014), and kernel based estimators (Shi et al., 2017). In the context of autoencoders learning implicit latent distribution can be seen as an adversarial game minimizing a specific divergence (Makhzani et al., 2015) or distance (Tolstikhin et al., 2017). Instead of adversarial formulations implicit distributions may also be learned directly by estimating the gradients of log density function using the Stein gradient estimator (Li & Turner, 2017). Finally, such gradient estimators can also be used to power variational inference with implicit posteriors enabling the use of posterior families with intractable densities (Shi et al., 2018).

Normalizing Flows:. Normalizing flows (NFs) (Rezende & Mohamed, 2015; Dinh et al., 2017) are a class of probabilistic models which use invertible transformations to map samples from a simple base distribution to samples from a more complex learned distribution. While there are many classes of normalizing flows (Papamakarios et al., 2019; Kobyzev et al., 2019), our work largely follows normalizing flows designed with partially-ordered dependencies, as found in affine coupling transformations (Dinh et al., 2017). Recently, normalizing flows have also been extended to Riemannian manifolds, such as spherical spaces in Gemici et al. (2016). In parallel to this work, normalizing flows have been extended to toriodal spaces (Rezende et al., 2020) and the data manifold (Brehmer & Cranmer, 2020). Finally, relying on affine coupling and GNNs, Liu et al. (2019a) develop graph normalizing flows (GNFs) for generating graphs. However, unlike our approach GNFs do not benefit from the rich geometry of hyperbolic spaces.

6. Conclusion

In this paper, we introduce two novel normalizing flows on hyperbolic spaces. We show that our flows are efficient to sample from, easy to invert and require only $\mathcal{O}(n)$ cost to compute the change in volume. We demonstrate the effectiveness of constructing hyperbolic normalizing flows for latent variable modeling of hierarchical data. We empirically observe improvements in structured density estimation, graph reconstruction and also generative modeling of tree-structured data, with large qualitative improvements in generated sample quality compared to Euclidean methods. One important limitation is in the numerical error introduced by clamping operations which prevent the creation of deep flow architectures. We hypothesize that this is an inherent limitation of the Lorentz model, which may be alleviated with newer models of hyperbolic geometry that use integer-based tiling (Yu & De Sa, 2019). In addition, while we considered hyperbolic generalizations of the coupling transforms to define our normalizing flows, designing new classes of invertible transformations like autoregressive and residual flows on non-Euclidean spaces is an interesting direction for future work.

Acknowledgements

Funding: AJB is supported by an IVADO Excellence Fellowship. RL was supported by Connaught International Scholarship and RBC Fellowship. WLH is supported by a Canada CIFAR AI Chair. This work was also supported by NSERC Discovery Grants held by WLH and PP. In addition the authors would like to thank Chinwei Huang, Maxime Wabartha, Andre Cianflone and Andrea Madotto for helpful feedback on earlier drafts of this work and Kevin Luk, Laurent Dinh and Niky Kamran for helpful technical discussions. The authors would also like to thank the anonymous reviewers for their comments and feedback and Aaron Lou, Derek Lim, and Leo Huang for catching a bug in the code.

References

Anderson, R. M., Anderson, B., and May, R. M. *Infectious diseases of humans: dynamics and control*. Oxford university press, 1992.

Brehmer, J. and Cranmer, K. Flows for simultaneous manifold learning and density estimation. *arXiv preprint arXiv:2003.13913*, 2020.

Burda, Y., Grosse, R., and Salakhutdinov, R. Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519*, 2015.

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

Chami, I., Ying, Z., Ré, C., and Leskovec, J. Hyperbolic graph convolutional neural networks. In *Advances in Neural Information Processing Systems*, pp. 4869–4880, 2019.

Davidson, T. R., Falorsi, L., De Cao, N., Kipf, T., and Tomczak, J. M. Hyperspherical variational auto-encoders. *arXiv preprint arXiv:1804.00891*, 2018.

De Cao, N., Titov, I., and Aziz, W. Block neural autoregressive flow. *arXiv preprint arXiv:1904.04676*, 2019.

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

Dinh, L., Sohl-Dickstein, J., and Bengio, S. Density estimation using real nvp. In *The 5th International Conference on Learning Representations (ICLR), Vancouver*, 2017.

Falorsi, L., de Haan, P., Davidson, T. R., and Forré, P. Reparameterizing distributions on lie groups. *arXiv preprint arXiv:1903.02958*, 2019.

Ganea, O., Bécigneul, G., and Hofmann, T. Hyperbolic neural networks. In *Advances in neural information processing systems*, pp. 5345–5355, 2018.

Gemici, M. C., Rezende, D., and Mohamed, S. Normalizing flows on riemannian manifolds. *arXiv preprint arXiv:1611.02304*, 2016.

Goh, K.-I., Cusick, M. E., Valle, D., Childs, B., Vidal, M., and Barabási, A.-L. The human disease network. *Proceedings of the National Academy of Sciences*, 104 (21):8685–8690, 2007.

Golomb, S. W. *Polyominoes: puzzles, patterns, problems, and packings*, volume 16. Princeton University Press, 1996.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Grathwohl, W., Chen, R. T., Bettencourt, J., Sutskever, I., and Duvenaud, D. Ffjord: Free-form continuous dynamics for scalable reversible generative models. *arXiv preprint arXiv:1810.01367*, 2018.

Grattarola, D., Livi, L., and Alippi, C. Adversarial autoencoders with constant-curvature latent manifolds. *Applied Soft Computing*, 81:105511, 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. *arXiv preprint arXiv:1805.09786*, 2018.

Hoffman, M. D., Blei, D. M., Wang, C., and Paisley, J. Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

Huang, C.-W., Krueger, D., Lacoste, A., and Courville, A. Neural autoregressive flows. In *Proceedings of the 35th international conference on Machine learning*, 2018.

Khrulkov, V., Mirvakhabova, L., Ustinova, E., Oseledets, I., and Lempitsky, V. Hyperbolic image embeddings. *arXiv preprint arXiv:1904.02239*, 2019.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kipf, T. N. and Welling, M. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016.

Kobyzev, I., Prince, S., and Brubaker, M. A. Normalizing flows: Introduction and ideas. *arXiv preprint arXiv:1908.09257*, 2019.

Law, M., Liao, R., Snell, J., and Zemel, R. Lorentzian distance learning for hyperbolic representations. In *International Conference on Machine Learning*, pp. 3672–3681, 2019.

Li, Y. and Turner, R. E. Gradient estimators for implicit models. *arXiv preprint arXiv:1705.07107*, 2017.

Liao, R., Li, Y., Song, Y., Wang, S., Hamilton, W., Duvenaud, D. K., Urtasun, R., and Zemel, R. Efficient graph generation with graph recurrent attention networks. In *Advances in Neural Information Processing Systems*, pp. 4257–4267, 2019.

Liu, J., Kumar, A., Ba, J., Kiros, J., and Swersky, K. Graph normalizing flows. In *Advances in Neural Information Processing Systems*, pp. 13556–13566, 2019a.

Liu, Q., Nickel, M., and Kiela, D. Hyperbolic graph neural networks. In *Advances in Neural Information Processing Systems*, pp. 8228–8239, 2019b.

Makhzani, A., Shlens, J., Jaitly, N., Goodfellow, I., and Frey, B. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015.

Mathieu, E., Le Lan, C., Maddison, C. J., Tomioka, R., and Teh, Y. W. Continuous hierarchical representations with poincaré variational auto-encoders. In *Advances in neural information processing systems*, pp. 12544–12555, 2019.

Nagano, Y., Yamaguchi, S., Fujita, Y., and Koyama, M. A wrapped normal distribution on hyperbolic space for gradient-based learning. In *International Conference on Machine Learning*, pp. 4693–4702, 2019.

Nickel, M. and Kiela, D. Poincaré embeddings for learning hierarchical representations. In *Advances in neural information processing systems*, pp. 6338–6347, 2017.

Nickel, M. and Kiela, D. Learning continuous hierarchies in the lorentz model of hyperbolic geometry. *arXiv preprint arXiv:1806.03417*, 2018.

Ovinnikov, I. Poincar\'e wasserstein autoencoder. *arXiv preprint arXiv:1901.01427*, 2019.

Papamakarios, G., Nalisnick, E., Rezende, D. J., Mohamed, S., and Lakshminarayanan, B. Normalizing flows for probabilistic modeling and inference. *arXiv preprint arXiv:1912.02762*, 2019.

Pennec, X. Intrinsic statistics on riemannian manifolds: Basic tools for geometric measurements. *Journal of Mathematical Imaging and Vision*, 25(1):127, 2006.

Ratcliffe, J. G. *Foundations of Hyperbolic Manifolds*. Number 149 in Graduate Texts in Mathematics. Springer-Verlag, 1994.

Rezende, D. J. and Mohamed, S. Variational inference with normalizing flows. In *Proceedings of the 32nd international conference on Machine learning*. ACM, 2015.

Rezende, D. J., Papamakarios, G., Racanière, S., Albergo, M. S., Kanwar, G., Shanahan, P. E., and Cranmer, K. Normalizing flows on tori and spheres. *arXiv preprint arXiv:2002.02428*, 2020.

Said, S., Bombrun, L., and Berthoumieu, Y. New riemannian priors on the univariate normal model. *Entropy*, 16 (7):4015–4031, 2014.

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

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2008.

Shi, J., Sun, S., and Zhu, J. Kernel implicit variational inference. *arXiv preprint arXiv:1705.10119*, 2017.

Shi, J., Sun, S., and Zhu, J. A spectral approach to gradient estimation for implicit distributions. *arXiv preprint arXiv:1806.02925*, 2018.

Skopek, O., Ganea, O.-E., and Bécigneul, G. Mixed-curvature variational autoencoders. *arXiv preprint arXiv:1911.08411*, 2019.

Tay, Y., Tuan, L. A., and Hui, S. C. Hyperbolic representation learning for fast and efficient neural question answering. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pp. 583–591, 2018.

Tolstikhin, I., Bousquet, O., Gelly, S., and Schoelkopf, B. Wasserstein auto-encoders. *arXiv preprint arXiv:1711.01558*, 2017.

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

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

Yang, Z., Cohen, W. W., and Salakhutdinov, R. Revisiting semi-supervised learning with graph embeddings. *arXiv preprint arXiv:1603.08861*, 2016.

Yu, T. and De Sa, C. M. Numerically accurate hyperbolic embeddings using tiling-based models. In *Advances in Neural Information Processing Systems*, pp. 2021–2031, 2019.