A Rotated Hyperbolic Wrapped Normal Distribution for Hierarchical Representation Learning

Seunghyuk Cho[1] **Juyong Lee**[1] **Jaesik Park**[1,2] **Dongwoo Kim**[1,2]

CSED POSTECH[1] **GSAI POSTECH**[2]

Abstract

We present a rotated hyperbolic wrapped normal distribution (RoWN), a simple yet effective alteration of a hyperbolic wrapped normal distribution (HWN). The HWN expands the domain of probabilistic modeling from Euclidean to hyperbolic space, where a tree can be embedded with arbitrary low distortion in theory. In this work, we analyze the geometric properties of the *diagonal* HWN, a standard choice of distribution in probabilistic modeling. The analysis shows that the distribution is inappropriate to represent the data points at the same hierarchy level through their angular distance with the same norm in the Poincaré disk model. We then empirically verify the presence of limitations of HWN, and show how RoWN, the proposed distribution, can alleviate the limitations on various hierarchical datasets, including noisy synthetic binary tree, WordNet, and Atari 2600 Breakout. The code is available at `https://github.com/ml-postech/RoWN`.

1 Introduction

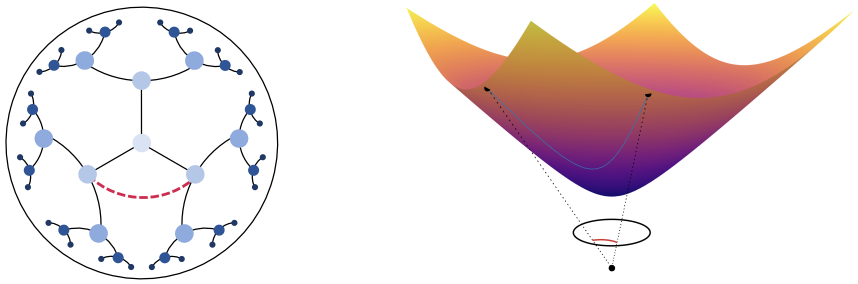
Hyperbolic space has served as an effective medium to learn parsimonious representations of hierarchical data, including vocabulary with relationships (Nickel and Kiela, 2017, 2018; Tifrea et al., 2019), knowledge graphs (Chami et al., 2020; Sun et al., 2020), and social networks (Zhao et al., 2011; Shavitt and Tankel, 2008). Recent studies reveal that the underlying anatomy in much complex data is non-Euclidean, supporting the success of representation learning in hyperbolic space (Bronstein et al., 2017). However, due to the absence of well-defined distribution that is easy to sample and has an analytic density function in hyperbolic space, earlier studies have focused on the non-probabilistic learning framework.

Hyperbolic wrapped normal distribution (HWN) has been recently proposed as an alternative to the celebrated normal distribution in Euclidean space. With an analytic density function and easiness of sampling, the HWN expands the domain of probabilistic modeling from Euclidean to hyperbolic space. The HWN has been successfully applied in various probabilistic models, including VAE (Kingma and Welling, 2014) and probabilistic word embeddings (Vilnis and McCallum, 2015). However, unlike the normal distribution in Euclidean space, the geometric characteristics of the HWN have not been fully understood so far. Therefore, figuring out what can be done or not with the HWN is difficult.

In this work, we analyze the geometric properties of the *diagonal* HWN, a standard choice of distribution in many probabilistic models (Mathieu et al., 2019; Nagano et al., 2019). Based on the observation that the principal axes of the diagonal normal distribution in Euclidean space have a parallel structure with the standard bases, we also focus on the structure of the principal axes of HWN in hyperbolic space to delve into a deeper understanding. Our analysis shows that the principal axes of the diagonal HWN are locally parallel to the standard bases in the Poincaré disk model.

Figure 1 (a) shows a common understanding on learned representation of hierarchical data in the Poincaré disk. The hierarchical structure spreads out like the spokes of a wheel. The local variation

36th Conference on Neural Information Processing Systems (NeurIPS 2022).



(a) The Poincaré disk model (b) The Lorentz model

Figure 1: Visualization of hyperbolic space models. (a) The Poincaré disk model can embed a given tree-structured data with low distortions as shown in the illustration. The black segments refer to the shortest path between the points in the Poincaré disk model. The red dashed line denotes the continuous points in the same hierarchy level. (b) The Lorentz model is another model for hyperbolic space. The blue line is the shortest path between the points on the Lorentz model. The red line is the projected geodesic via the diffeomorphism, which is still a geodesic of the Poincaré disk model.

in a hierarchy can then be represented as an angular difference between nodes at the same level of the norm, i.e., the principal axis representing local variation is orthogonal to the radial axis. However, our analysis of the geometric property reveals that the local variation can only be modeled along with the standard bases with the diagonal HWN.

To fix the structure of the principal axes in the diagonal HWN, we propose a simple yet effective alteration of HWN, named a rotated hyperbolic wrapped normal distribution (RoWN). By rotating the diagonal covariance matrix before parallel transportation of HWN, we could resolve the limitations in local variation structure while keeping the valuable properties, such as easy sampling and tractable density of the original HWN.

We verify the representation learned with RoWN agrees with the common characteristic of representation in hyperbolic space, which is barely observable with the diagonal and the full covariance HWN, by using *synthetic noisy binary tree* dataset. We demonstrate the usefulness of RoWN on the benchmark datasets: WordNet and Atari 2600 breakout. We summarize our contributions as follows:

1. We provide an analysis of the geometric properties of HWN and its potential limitations in representation learning.
2. We propose a novel and efficient method of using hyperbolic distribution, namely RoWN, and apply it to probabilistic models.
3. We demonstrate the performance of RoWN through the comparison with the Euclidean normal distribution, diagonal covariance HWN, and full covariance HWN on one synthetic dataset and two benchmark datasets.

2 Preliminaries

This section reviews the wrapped normal distribution defined on the Lorentz model. We first introduce the Lorentz model of hyperbolic space and necessary concepts to understand the wrapped normal distribution.

2.1 The Lorentz model

Hyperbolic space is a non-Euclidean space, having a constant negative Gaussian curvature. Figure 1 illustrates two models for hyperbolic space, among four standard equivalent models: (1) Klein model, (2) the Poincaré disk model, (3) the Lorentz (hyperboloid) model, and (4) the Poincaré half-plane model. In particular, the Lorentz model is famous for its numerical stability in computing the distance and comes with a simpler closed form of geodesics (Nickel and Kiela, 2018). The Lorentz model $\mathbb{L}^n$ is the Riemannian manifold consisting of the set of points $\boldsymbol{z} \in \mathbb{R}^{n+1}$ satisfying $\langle \boldsymbol{z}, \boldsymbol{z} \rangle_{\mathcal{L}} = -1$ and

$z_0 > 0$, where the Lorentizan product $\langle \cdot, \cdot \rangle_{\mathcal{L}}$ is defined as:

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle_{\mathcal{L}} := -x_0 y_0 + \sum_{i=1}^{n} x_i y_i,$$

which also works as the metric tensor on hyperbolic space, i.e., the metric tensor g of the Lorentz model is $g(\boldsymbol{x}) = \mathrm{diag}[-1, 1, \cdots, 1]$

2.2 Tangent space of the Lorentz model

We denote the tangent space of $\boldsymbol{x} \in \mathbb{L}^n$ as $\mathcal{T}_{\boldsymbol{x}}\mathbb{L}^n$, which is a set of points satisfying the orthogonality relation with $\boldsymbol{x}$ in terms of the Lorentzian product: $T_{\boldsymbol{x}}\mathbb{L}^n := \{\boldsymbol{u} : \langle \boldsymbol{u}, \boldsymbol{x} \rangle_{\mathcal{L}} = 0\}$. The metric tensor g induces an inner product of two tangent vectors from a tangent space. Geodesic $\gamma : [0, 1] \to \mathbb{L}^n$ generalizes straight lines in the Riemannian manifold which is the shortest curve between two points. The exponential map $\exp_{\boldsymbol{x}} : \mathcal{T}_{\boldsymbol{x}}\mathbb{L}^n \to \mathbb{L}^n$ maps a tangent vector $\boldsymbol{u} \in \mathcal{T}_{\boldsymbol{x}}\mathbb{L}^n$ onto $\mathbb{L}^n$ as:

$$\exp_{\boldsymbol{x}}(\boldsymbol{u}) := \cosh(\|\boldsymbol{u}\|_{\mathcal{L}})\boldsymbol{x} + \sinh(\|\boldsymbol{u}\|_{\mathcal{L}})\frac{\boldsymbol{u}}{\|\boldsymbol{u}\|_{\mathcal{L}}}, \tag{1}$$

such that $\exp_{\boldsymbol{x}}(\boldsymbol{u}) = \boldsymbol{y}, \gamma(0) = \boldsymbol{x}, \gamma(1) = \boldsymbol{y}$. The log map, inverse of the exponential map, is defined as $\log_{\boldsymbol{x}}(\boldsymbol{y}) := \exp_{\boldsymbol{x}}^{-1}(\boldsymbol{u})$.

Parallel transport is an operation that transports a tangent vector in the tangent space at $\boldsymbol{x}$ to another vector in the tangent space at $\boldsymbol{y}$ along the geodesic from $\boldsymbol{x}$ to $\boldsymbol{y}$ without losing the parallel property. The parallel transport in the Lorentz model is given by:

$$\mathrm{PT}_{\boldsymbol{x} \to \boldsymbol{y}}(\boldsymbol{v}) := \boldsymbol{v} + \frac{\langle \boldsymbol{y} - \alpha \boldsymbol{x}, \boldsymbol{v} \rangle_{\mathcal{L}}}{\alpha + 1}(\boldsymbol{x} + \boldsymbol{y}), \tag{2}$$

where $\alpha = -\langle \boldsymbol{x}, \boldsymbol{y} \rangle_{\mathcal{L}}$.

2.3 Hyperbolic wrapped normal distribution

One of the key challenges in adopting hyperbolic space to probabilistic models is finding a distribution on hyperbolic space that is easy to sample and has a closed-form density function. The two most common distributions of hyperbolic space used in previous work are Riemannian normal distribution (Mathieu et al., 2019; Said et al., 2014) and hyperbolic wrapped normal distribution (HWN) (Nagano et al., 2019). Our work is mainly based on the HWN because sampling in Riemannian normal distribution is limited with only a unit variance.

The sampling process with the HWN follows:

1. Sample $\boldsymbol{v} \in \mathbb{R}^n$ from a Gaussian distribution $\mathcal{N}(\mathbf{0}, \Sigma)$ in Euclidean space.
2. Parallel transport the vector $[0, \boldsymbol{v}] \in \mathcal{T}_{\mathbf{0}_{\mathcal{L}}}\mathbb{L}^n$ to the tangent space.
3. Project the transported tangent vector to $\mathbb{L}^n$ using exponential mapping.

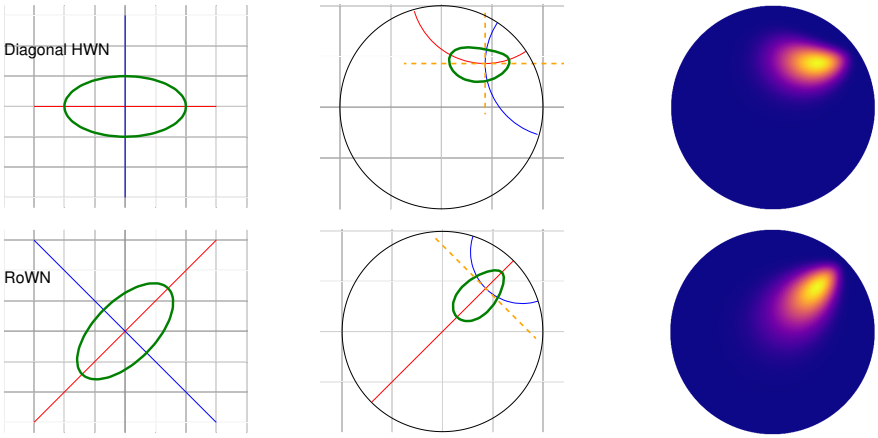
The density of a sample can be measured via the change of variable method. For convenience, we denote the procedures 2 and 3 as a single operation:

$$f_{\boldsymbol{\mu}}(\boldsymbol{v}) := \exp_{\boldsymbol{\mu}}(\mathrm{PT}_{\mathbf{0}_{\mathcal{L}} \to \boldsymbol{\mu}}([0, \boldsymbol{v}])), \tag{3}$$

with given $\boldsymbol{\mu} \in \mathbb{L}^n$ and $\boldsymbol{v} \in \mathbb{R}^n$, and $\mathbf{0}_{\mathcal{L}} = [1, \ldots, 0]$ stands for the origin of $\mathbb{L}^n$.

3 Rotated Hyperbolic Wrapped Normal Distribution

This section introduces several observations on the geometric properties of the HWN distribution transformation from the Euclidean space to hyperbolic space. We show the limitations of the diagonal HWN for representation learning of hierarchical data. Then, we propose a simple yet effective modification of the HWN, called a rotated hyperbolic wrapped normal distribution.



(a) (b) (c)

Figure 2: Visualization of (a) the principal axes of the normal distribution in the Euclidean space, (b) the transported version of two axes in hyperbolic space, and (c) the probability density plot of the distributions. (a) The contour line (green) can be represented as an ellipse with major principal axis (red) and minor principal axis (blue). (b) The transformed principal axes become geodesics in hyperbolic space. The major principal axis of the RoWN passes the hyperbolic origin and crosses with the minor axis at the mean point, whereas the major and minor axes of the diagonal HWN is locally parallel to the standard axes. (c) The principal axes determine the shapes of the variational distributions.

3.1 Observations on the hyperbolic wrapped normal distribution

First, we investigate the changes in the normal distribution during the transformation from Euclidean to hyperbolic space by Equation 3. As the principal axes characterize the covariance structure of the normal distribution in the Euclidean space, we investigate how the principal axes are transformed in hyperbolic space. Before deriving our main proposition, we first show that the straight lines that pass through the origin in the Euclidean space are transformed to the geodesics in hyperbolic space by Equation 3.

Proposition 1. *Suppose $\ell_{\boldsymbol{s}}(t) = t\boldsymbol{s} \in \mathbb{R}^n$ be a line passing through the origin, where $\boldsymbol{s} \in \mathbb{R}^n$ is a directional vector. Then the curve $f_{\boldsymbol{\mu}}(\ell_{\boldsymbol{s}}(t))$ in the Lorentz model $\mathbb{L}^n$ becomes a geodesic.*

The proof of the proposition is provided in Appendix A. The proposition indicates that every straight line that passes through the origin including the principal axes, is transformed into a geodesic in hyperbolic space.

Based on the first proposition, we provide our main proposition, which fully characterizes the structure of principal axes when projected to the Poincaré disk model:

Proposition 2. *Define $\mathrm{Proj}(\boldsymbol{u})$ to be the projection function from the Lorentz model to the Poincaré model, i.e., $\mathrm{Proj}(\boldsymbol{u}) = \frac{x_{1:}(\boldsymbol{u})}{x_0(\boldsymbol{u})+1}, \forall \boldsymbol{u} \in \mathbb{L}^n$. If $\ell_{\boldsymbol{s}}$ is a principal axis of the normal distribution defined in $\mathbb{R}^n$ and $\boldsymbol{\mu}$ the mean of HWN in $\mathbb{L}^n$, then $\boldsymbol{s}$ is the tangent vector of $\mathrm{Proj}(f_{\boldsymbol{\mu}}(\ell_{\boldsymbol{s}}))$ on $\mathrm{Proj}(\boldsymbol{\mu})$.*

The proof of the proposition is provided in Appendix B. The proposition reveals that the principal axes of the HWN are locally parallel to the standard bases in the Poincaré disk model. To visualize the proposition, we plot the contour line and the principal axes of the two-dimensional diagonal normal distribution before and after the transformation in Figure 2. We observe that the tangent lines of the transformed principal axes are parallel to the standard bases in hyperbolic space.

When a popular diagonal normal distribution is employed as a variational distribution, the locally parallel principal axes might be problematic in learning hierarchical representations. For example, suppose one tries to represent the variability along the radial direction or in angular differences. In the case, both the major (red) and minor (blue) axes in Figure 2b cannot model the variability properly.

Algorithm 1 Sampling process with the rotated hyperbolic wrapped normal distribution

Input Mean $\boldsymbol{\mu} \in \mathbb{L}^n$, diagonal covariance matrix $\Sigma \in \mathbb{R}^{n \times n}$

Output Sample $\boldsymbol{z} \in \mathbb{L}^n$

$\boldsymbol{x} = [\pm 1, \ldots, 0] \in \mathbb{R}^n, \boldsymbol{y} = \boldsymbol{\mu}_{1:} / \|\boldsymbol{\mu}_{1:}\|$ ▷ $\pm$ is determined by the sign of $\boldsymbol{\mu}_0$

$\boldsymbol{R} = \boldsymbol{I} + (\boldsymbol{y}^T\boldsymbol{x} - \boldsymbol{x}^T\boldsymbol{y}) + (\boldsymbol{y}^T\boldsymbol{x} - \boldsymbol{x}^T\boldsymbol{y})^2/(1 + \langle \boldsymbol{x}, \boldsymbol{y} \rangle)$

Rotate $\hat{\Sigma} = \boldsymbol{R}\Sigma\boldsymbol{R}^T$

Sample $\boldsymbol{v} \sim \mathcal{N}(\boldsymbol{0}, \hat{\Sigma})$

return $\boldsymbol{z} = f_{\boldsymbol{\mu}}(\boldsymbol{v})$

3.2 Rotated hyperbolic wrapped normal distribution

Based on the observation, we propose a simple yet effective alternative to the diagonal HWN, a rotated hyperbolic wrapped normal distribution (RoWN). Rotating the covariance matrix to the direction of $\boldsymbol{\mu}$ enables aligning the major axis of the normal distribution in the Euclidean space to the radial axis in hyperbolic space as visualized in the Figure 2.

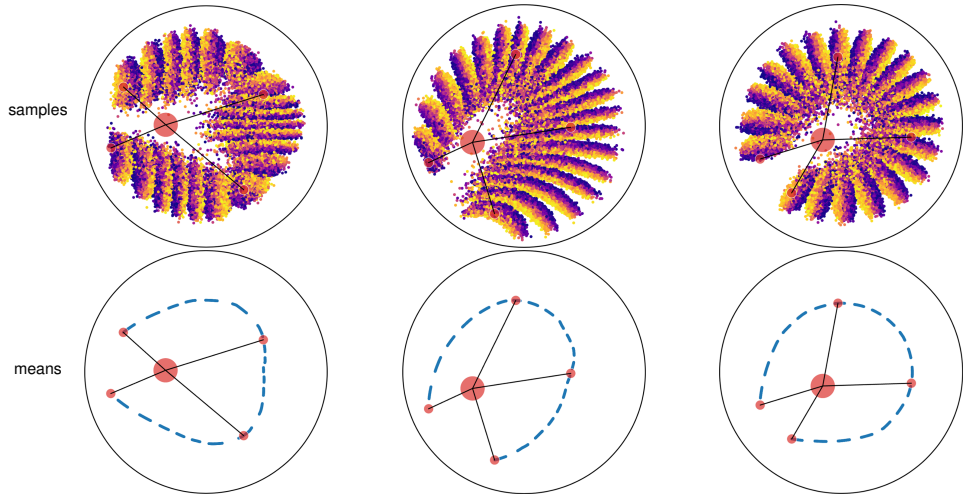
To construct the distribution, we start with a mean vector $\boldsymbol{\mu} \in \mathbb{L}^n$ and a diagonal covariance matrix Σ as in the standard HWN. We change the covariance matrix of the normal distribution as follows:

1. Compute the rotation matrix $\boldsymbol{R}$ that rotates the x-axis ($[\pm 1, \ldots, 0] \in \mathbb{R}^n$) to $\boldsymbol{\mu}_{1:}$.
2. Substitute the covariance matrix of Gaussian normal with $\boldsymbol{R}\Sigma\boldsymbol{R}^T$.

Thus, the rotation matrix $\boldsymbol{R}$, which rotates a unit vector from $\boldsymbol{x}$ to $\boldsymbol{y}$, can be computed as:

$$\boldsymbol{R} = \boldsymbol{I} + (\boldsymbol{y}^T\boldsymbol{x} - \boldsymbol{x}^T\boldsymbol{y}) + \frac{1}{1 + \langle \boldsymbol{x}, \boldsymbol{y} \rangle}(\boldsymbol{y}^T\boldsymbol{x} - \boldsymbol{x}^T\boldsymbol{y})^2. \tag{4}$$

The pseudo-code of the sampling process of RoWN is in Algorithm 1. Note that the construction is straightforward but still keeps the following benefits of the HWN: 1) The sampling can be done efficiently, and 2) the computation of the probability density of the samples is tractable. As the HWN provides a tractable probability density function for any kind of covariance matrix, we can easily compute the probability density of a given sample from RoWN. See Appendix C for more details.



(a) Diagonal HWN (b) Full covariance HWN (c) RoWN

Figure 3: Visualization of the variational distribution of hyperbolic VAE on a synthetic binary tree dataset with different variational distributions. The red dots denote the variational means of the root and four representative children. The upper row shows the samples from the variational distributions where the color denotes the level of noise. The bottom row shows the means of the variational distributions. Overall, RoWN better aligns local variation in angular difference.

We empirically demonstrate the influence of different variational distributions in Figure 3. We visualize the variational distributions of the synthetic binary tree dataset of depth two after training a hyperbolic VAE. As the result shows, the model with the diagonal HWN represents the variation in a child parallel to the standard bases, whereas the model with RoWN represents the variation in angular difference. Please check the detailed description on the synthetic dataset in Section 4.2.

4 Experiments

In this section, we first explain the two applications of the distribution defined on hyperbolic space: hyperbolic VAE and probabilistic hyperbolic word embedding model. We then conduct three different experiments to compare the performance of RoWN with four baselines, including the Gaussian distribution in the Euclidean space, the isotropic HWN (Nagano et al., 2019), the diagonal HWN, and the full covariance HWN. We also provide an additional study on a variant of RoWN with learnable rotation direction $\boldsymbol{y}$ in Algorithm 1, and the results are in Table 9. The details of the experiments are described in Appendix D.

4.1 Applications of the hyperbolic distribution

Hyperbolic VAE. The hyperbolic VAE, whose latent space is hyperbolic space, has been shown to be efficient for capturing the hierarchical structure of the data (Nagano et al., 2019; Mathieu et al., 2019). The evidence lower bound of the hyperbolic VAE can be written as:

$$\mathcal{L}_{\mathrm{ELBO}}(\theta, \phi) := \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})\cdot\sqrt{\det(g)}}[\log p_\theta(\boldsymbol{x} \mid \boldsymbol{z})] - D_{\mathrm{KL}}\left(q_\phi(\boldsymbol{x} \mid \boldsymbol{z}) \cdot \sqrt{\det(g)} \parallel p(\boldsymbol{z}) \cdot \sqrt{\det(g)}\right),$$

where q_ϕ is the encoder, p_θ is the decoder, g is the metric tensor of the chosen model of hyperbolic space, and $p(\boldsymbol{z})$ is a prior distribution. The distributions defined on hyperbolic spaces, such as HWN and RoWN, are used to define encoder $q_\phi(\boldsymbol{z} \mid \boldsymbol{x}) \cdot \sqrt{\det(g)}$ and prior $p(\boldsymbol{z}) \cdot \sqrt{\det(g)}$. In hyperbolic VAEs, due to the absence of the closed-form KL divergence in the hyperbolic distributions, the KL divergence between the encoder and the prior is usually approximated with Monte-Carlo sampling (Nagano et al., 2019; Mathieu et al., 2019).

Probabilistic hyperbolic word embedding model. The probabilistic word embedding models aim to learn probabilistic representations of words (Vilnis and McCallum, 2015; Nagano et al., 2019; Tifrea et al., 2019). The embeddings learned on hyperbolic spaces have shown better performance than the Euclidean counterpart (Nagano et al., 2019; Tifrea et al., 2019). Given the hypernymy relationships between the words, the probabilistic hyperbolic embedding model learns the probabilistic representation of words by minimizing the following objective:

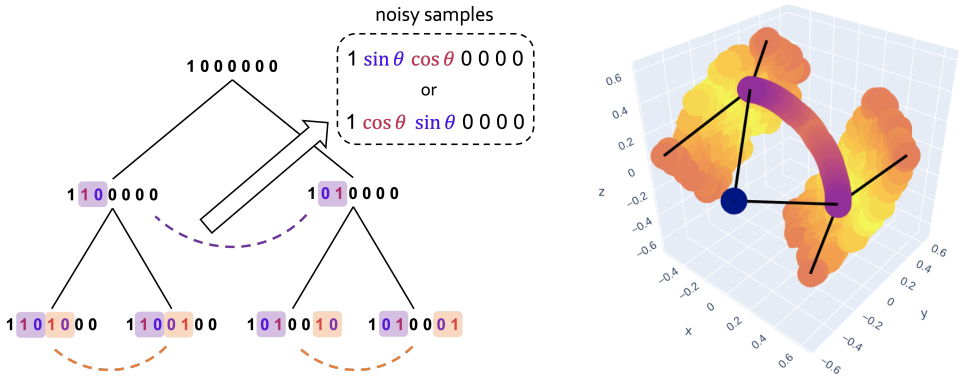
$$\mathcal{L}_{\mathrm{word}}(\boldsymbol{\theta}) := \mathbb{E}_{(s\sim t, s\not\sim t')}[\max(0, m + D_{\mathrm{KL}}(q_s \parallel q_t) - D_{\mathrm{KL}}(q_s \parallel q_{t'}))], \tag{5}$$

where m is a margin, q_i is a distribution for word i parameterized by θ, and $s \sim t$ and $s \not\sim t$ denote the presence and absence of hypernymy relation between word pair s and t, respectively.

4.2 Noisy synthetic binary tree

A synthetic binary tree dataset is first used to show the performance of representing hierarchy in Nagano et al. (2019), where each node in a tree corresponds to a sequence of binary values. Figure 4a shows an example of the depth three binary tree, where a parent and child only differ in one digit. We add spherical noises to the nodes in the same level of hierarchy as described in Figure 4a as the noisy samples. With the noisy samples, we can create a dataset containing a local-level variation in the hierarchy. For the experiments, we uniformly sample the spherical noise from $[0, \pi/4]$.

We train hyperbolic VAE on *noisy synthetic binary tree* with varying depths. The detailed model description is available in Appendix D.1. We set the latent dimension the same as the depth. We report 1) the correlation between the hamming distance and the embedding distance and 2) the correlation between the depth and the Poincaré norm of the embeddings. The first correlation is computed over all possible pairs of test points. As Table 1 shows, the full covariance HWN and RoWN improve the diagonal HWN except depth six, outperforming the Euclidean model in every setting. RoWN preserves the depth information better than the other distributions in general. We additionally visualize the variational mean obtained by training the tree of depth three in Figure 4b, where the hierarchical structure is well preserved in the hyperbolic embedding space.



(a) Noisy synthetic binary tree

(b) Learned representations

Figure 4: Illustration of a noisy synthetic binary tree. (a) We construct a noisy synthetic binary tree by adding spherical noises defined with θ. The continuous samples are generated at the same distance from the root. (b) We train the depth three noisy synthetic binary tree with hyperbolic VAE with RoWN as a variational distribution and visualize the means of the variational distributions. The black lines show the underlying hierarchical structure, and the color denotes the level of noise. The hierarchy and the local variations are well preserved in the representations.

4.3 WordNet

We train a probabilistic word embedding model with WordNet dataset (Fellbaum, 1998), which consists of 82,115 nouns and 743,241 hypernymy relationships. We have initialized the embeddings from $\mathcal{N}(0, 0.01I)$, which are then moved to the Lorentz model using the exponential map. We use the learning rate warm-up proposed in (Nagano et al., 2019). We evaluate the learned representations by computing the average rank of all the hypernymies. The rank of a given pair of words s and t is computed among the distances between all possible pairs of the words s and t' without hypernymy. Table 2 shows the empirical performances of representing the word data. We report the performance with the mean rank (MR) and the mean average precision (mAP). RoWN preserves the hierarchical structure better than the other distributions, while the full covariance HWN often performs worse than RoWN.

Failure of the full covariance HWN. In theory, the full covariance HWN needs to have at least a similar performance to RoWN since the RoWN is the particular case of the full covariance. The

Table 1: Results of *noisy synthetic binary tree*. The results are averaged over 10 runs. The hyperbolic models outperform the Euclidean model in all settings. Overall, RoWN preserves the hierarchical information better than the other distributions.

		Depth			
		4	5	6	7
Correlation w/ distance	Euclidean	$0.748_{\pm.032}$	$0.740_{\pm.013}$	$0.741_{\pm.008}$	$0.733_{\pm.014}$
	HWN (isotropic Σ)	$0.773_{\pm.030}$	$0.809_{\pm.016}$	$0.798_{\pm.008}$	$0.735_{\pm.022}$
	HWN (diagonal Σ)	$0.814_{\pm.008}$	$0.791_{\pm.023}$	$0.817_{\pm.010}$	$0.759_{\pm.025}$
	HWN (full Σ)	$0.827_{\pm.015}$	$0.798_{\pm.026}$	$0.798_{\pm.010}$	$0.794_{\pm.014}$
	RoWN	$0.820_{\pm.015}$	$0.807_{\pm.017}$	$0.822_{\pm.017}$	$0.788_{\pm.016}$
Correlation w/ depth	Euclidean	$0.762_{\pm.117}$	$0.807_{\pm.038}$	$0.712_{\pm.054}$	$0.612_{\pm.049}$
	HWN (isotropic Σ)	$0.902_{\pm.033}$	$0.867_{\pm.034}$	$0.811_{\pm.029}$	$0.602_{\pm.066}$
	HWN (diagonal Σ)	$0.918_{\pm.028}$	$0.808_{\pm.076}$	$0.862_{\pm.035}$	$0.697_{\pm.076}$
	HWN (full Σ)	$0.956_{\pm.015}$	$0.878_{\pm.051}$	$0.870_{\pm.033}$	$0.815_{\pm.055}$
	RoWN	$0.930_{\pm.026}$	$0.911_{\pm.027}$	$0.901_{\pm.034}$	$0.827_{\pm.047}$

Table 2: Results of *WordNet*. The results are an average of 5 runs. Based on the rank of hypernymy pairs among non-hypernymy pairs, we report the mean rank (MR) and mean average precision (mAP) for evaluation.

		Latent dimension		
		5	10	20
MR	Euclidean	$13.968_{\pm 0.504}$	$3.862_{\pm 0.281}$	$1.955_{\pm 0.157}$
	HWN (isotropic Σ)	$14.568_{\pm 2.203}$	$4.470_{\pm 0.669}$	$3.125_{\pm 0.455}$
	HWN (diagonal Σ)	$16.590_{\pm 1.146}$	$3.891_{\pm 0.447}$	$2.062_{\pm 0.088}$
	HWN (full Σ)	$557.309_{\pm 18.006}$	$466.513_{\pm 75.142}$	$599.140_{\pm 18.916}$
	RoWN	$16.271_{\pm 2.985}$	$2.888_{\pm 0.162}$	$1.783_{\pm 0.090}$
mAP	Euclidean	$0.565_{\pm 0.014}$	$0.801_{\pm 0.020}$	$0.902_{\pm 0.008}$
	HWN (isotropic Σ)	$0.617_{\pm 0.012}$	$0.820_{\pm 0.013}$	$0.847_{\pm 0.017}$
	HWN (diagonal Σ)	$0.565_{\pm 0.020}$	$0.805_{\pm 0.015}$	$0.905_{\pm 0.007}$
	HWN (full Σ)	$0.032_{\pm 0.003}$	$0.063_{\pm 0.005}$	$0.079_{\pm 0.021}$
	RoWN	$0.593_{\pm 0.024}$	$0.844_{\pm 0.009}$	$0.921_{\pm 0.005}$

performance of the full covariance HWN often performs worse with the probabilistic hyperbolic word embedding models than the hyperbolic VAEs. We speculate that reason is because of the relatively simple prior in the hyperbolic VAE, whereas the probabilistic word embedding models need to compute the KL divergence between the full covariance HWNs. Our additional experiments reported in Table 11 confirm our speculation as the number of training samples for the KL divergence increases, the performance increases slightly.

Discussion of the root placement. Note that due to the isometry of the geometry, the root node can be placed anywhere in hyperbolic space. In other words, infinitely many sets of embeddings preserve the same pairwise distances between the nodes. To place the root near the origin, we initialize all embeddings with the zero mean distribution as done in (Nagano et al., 2019). We find that this initialization helps the root placed near the origin. A study on the effects of the initialization methods is shown in Table 3. Further details about the root placement can be found in Appendix E.

4.4 Atari 2600 Breakout

Trajectories of some Atari 2600 games can be structured as a tree-like hierarchy along the time horizon as Nagano et al. (2019) points out. For example, given Atari 2600 Breakout, the root node can be the starting state of the game where no blocks have been broken. As the game progresses the states of the blocks form a hierarchical structure depending on which blocks have been broken. To

Table 3: The effects of initializations. We test different initializations on the deterministic hyperbolic word embedding models trained with the subset of the WordNet dataset. The `near zero vector` model initializes the embeddings with uniform distribution $\mathcal{U}(-0.001, 0.001)$, while the `near one vector` model initializes the embeddings with uniform distribution $\mathcal{U}(0.999, 1.001)$. While the Poincaré norm of the root node differs, the other metrics remain similar.

		Latent dimension			
		2	5	10	20
MR	Near zero vector	$4.346_{\pm .643}$	$3.270_{\pm .144}$	$2.828_{\pm .098}$	$2.508_{\pm .049}$
	Near one vector	$4.029_{\pm .415}$	$3.209_{\pm .094}$	$2.856_{\pm .075}$	$2.491_{\pm .051}$
mAP	Near zero vector	$0.821_{\pm .016}$	$0.891_{\pm .006}$	$0.891_{\pm .005}$	$0.895_{\pm .003}$
	Near one vector	$0.829_{\pm .010}$	$0.894_{\pm .004}$	$0.891_{\pm .004}$	$0.896_{\pm .003}$
The Poincaré norm of the root node	Near zero vector	$0.122_{\pm .036}$	$0.042_{\pm .015}$	$0.031_{\pm .007}$	$0.024_{\pm .004}$
	Near one vector	$0.505_{\pm .076}$	$0.650_{\pm .010}$	$0.732_{\pm .003}$	$0.801_{\pm .001}$

Table 4: Results of *Atari 2600 Breakout*. The results are averaged over 5 runs. We measure the correlation between the score of an image and the Poincaré norm of the variational mean. The HWN with the isotropic covariance can be viewed as a variant of RoWN.

		Latent dimension		
		10	15	20
Correlation btw. score and norm	Euclidean	$0.379_{\pm.007}$	$0.436_{\pm.029}$	$0.479_{\pm.020}$
	HWN (isotropic Σ)	$0.513_{\pm.012}$	$0.598_{\pm.021}$	$0.607_{\pm.015}$
	HWN (diagonal Σ)	$0.478_{\pm.011}$	$0.513_{\pm.006}$	$0.513_{\pm.008}$
	HWN (full Σ)	$0.483_{\pm.011}$	$0.520_{\pm.009}$	$0.563_{\pm.010}$
	RoWN	$0.497_{\pm.014}$	$0.556_{\pm.014}$	$0.561_{\pm.029}$

learn the implicit hierarchy that can be observed from the trajectories of Breakout, we train the VAE models with the Atari 2600 Breakout images.

The images of Breakout are collected by using a pre-trained Deep Q-network (Mnih et al., 2015) and divided into training set and test set with 90,000 and 10,000 images respectively. We label each image with the score obtained from the game environment. So the labels are correlated to the number of broken blocks. To train VAE, we use a DCGAN-based architecture, which was originally used to evaluate HWN in Nagano et al. (2019). The detailed architecture is provided in Appendix D.4. To evaluate the models, we measure the correlation between the Poincaré norm of the test images and the labeled scores. The evaluation results and the generated images from the trained models are reported in Table 4 and Figure 5, respectively.

In learning Breakout images, RoWN and the full covariance HWN outperform the diagonal HWN. The isotropic HWN, where the covariance matrix is invariant to any rotation matrix, shows a better correlation than the others in all the settings. However, as reported in Table 14, the test ELBO values are relatively worse than the others. While the isotropic HWN shows a high correlation but relatively lower test ELBO, RoWN shows competitive test ELBO to the others and well aligns the hierarchical structures with respect to the norm in low latent dimensions.

5 Related work

Hyperbolic space for hierarchical representation learning. Earlier studies on the hierarchical representation learning have focused on modeling explicit hierarchical structures through the Bayesian non-parametrics (Griffiths et al., 2003; Larsen et al., 2002; Salakhutdinov et al., 2012; Teh et al., 2007; Ghahramani et al., 2010; Heller and Ghahramani, 2005) or embedding the hierarchical structure into Euclidean space (Nickel et al., 2011; Grover and Leskovec, 2016; Nguyen et al., 2017). Euclidean

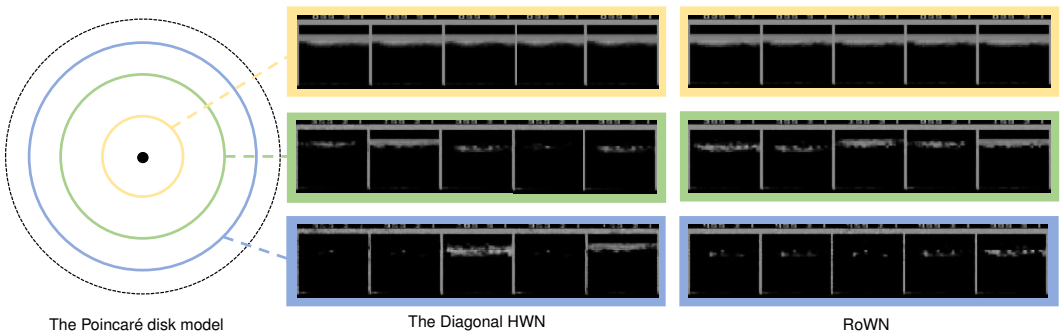


Figure 5: The generated images from a trained VAE endowed with RoWN by using Atari 2600 Breakout images. Every five images are generated from the randomly sampled latent vectors of dimensionality two having the Poincaré norm 0.1, 0.9, 0.95. The larger norm of the sample latent vectors, the more blocks are broken out in the generated images.

space is later shown to require an excessive number of dimensions to embed the original hierarchical structure without any distortion (Linial et al., 1994).

Hyperbolic space has been proposed as an alternative medium to embed the hierarchical data. Theoretically, any tree-structured data can be embedded in hyperbolic space with arbitrarily low distortion (Sarkar, 2011). Based on the theory, learning representation of hierarchical structure in hyperbolic space has been shown great success in various datasets including WordNet hierarchy, graph-structured data, and social network data (Nickel and Kiela, 2017, 2018; Chami et al., 2020; Sun et al., 2020; Zhao et al., 2011; Shavitt and Tankel, 2008). However, these studies are only limited to modeling the explicit hierarchy, which means that the dataset contains an explicit relation between data points. Furthermore, the learning frameworks are limited in the non-probabilistic setting because of the absence of well-behaved distribution in hyperbolic space.

Distributions in hyperbolic space. The probabilistic model enables measuring the uncertainty and provides a principled way of learning. To extend the probabilistic learning framework from Euclidean to hyperbolic space or Riemannian manifold in general, one needs to define a well-behaved distribution that has a tractable density and is easy to sample from. Recently, a few studies have proposed probabilistic learning schemes in hyperbolic space (Mathieu et al., 2019; Nagano et al., 2019). Mathieu et al. (2019) introduces parametrizable sampling schemes for the two canonical Gaussian generalizations defined on the Poincaré disk model. The scheme is used to train a hyperbolic VAE and show an improved generalization performance with high interpretability. Nagano et al. (2019) suggests a method of integrating the Bayesian framework with hyperbolic space in the Lorentz model where the simpler closed form of geodesics is defined. Normalizing flow (Rezende and Mohamed, 2015) can be also used to define the hyperbolic distribution in the probabilistic learning framework. Bose et al. (2020) propose two normalizing flows defined on hyperbolic space, which show improvements in learning hierarchical structures in graph data. Mathieu and Nickel (2020) elevate the continuous normalizing flow (Chen et al., 2018) defined on the Euclidean space to arbitrary Riemannian manifold, including the hyperbolic space. Based on Nagano et al. (2019), we analyze the geometric properties of the distribution lying on the Lorentz model and show the limitations of the existing method.

6 Limitations

We explore a better method of representing hierarchical data in hyperbolic space. To this end, we propose a simple yet effective alternative of hyperbolic wrapped normal distribution. However, the proposed distribution is limited only to hyperbolic space, and no generalization method for Riemannian space is studied yet. To explore the usefulness of alternative Riemannian spaces, finding a common distribution that can work well in any Riemannian space will be necessary.

RoWN is a subset of the full covariance HWN. However, in many cases, RoWN outperforms the full covariance HWN in our experiments. In general, optimizing the covariance matrix requires learning the quadratic number of parameters with respect to the dimensionality. We conjecture the hardness of optimization leads to the poor performance of the full covariance HWN. To overcome this limitation, a search for a better optimization algorithm in hyperbolic space needs to be explored.

7 Conclusions

In this work, we propose a novel method of using RoWN for representing the data with a hierarchical structure. With an in-depth analysis of the geometric properties of HWN, we demonstrate why the common choice of the diagonal covariance matrix for HWN may be inappropriate but the rotated covariance matrix. Our empirical results present that RoWN exhibits better representation ability, both qualitatively and quantitatively, compared to all the baselines: Euclidean normal distribution, diagonal HWN, and full covariance HWN. We hope that our method helps better understanding the anatomy of hyperbolic space and be a promising technique for efficient representation learning.

Acknowledgement

This work was partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2019-0-01906, Artificial Intelligence Graduate School Program (POSTECH)) and the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (NRF-2021R1C1C1011375).

References

Joey Bose, Ariella Smofsky, Renjie Liao, Prakash Panangaden, and Will Hamilton. Latent variable modelling with hyperbolic normalizing flows. In *International Conference on Machine Learning*, 2020.

Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 2017.

Ines Chami, Adva Wolf, Da-Cheng Juan, Frederic Sala, Sujith Ravi, and Christopher Ré. Low-dimensional hyperbolic knowledge graph embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, July 2020.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Christiane Fellbaum. *WordNet: An Electronic Lexical Database*. Bradford Books, 1998.

Zoubin Ghahramani, Michael Jordan, and Ryan P Adams. Tree-structured stick breaking for hierarchical data. *Advances in neural information processing systems*, 23, 2010.

Thomas Griffiths, Michael Jordan, Joshua Tenenbaum, and David Blei. Hierarchical topic models and the nested chinese restaurant process. *Advances in neural information processing systems*, 16, 2003.

Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 2016.

Katherine A Heller and Zoubin Ghahramani. Bayesian hierarchical clustering. In *Proceedings of the 22nd international conference on Machine learning*, pages 297–304, 2005.

Daniella Horan, Eitan Richardson, and Yair Weiss. When is unsupervised disentanglement possible? *Advances in Neural Information Processing Systems*, 34:5150–5161, 2021.

Jaemin Jo and Jinwook Seo. Disentangled representation of data distributions in scatterplots. In *2019 IEEE Visualization Conference (VIS)*, pages 136–140. IEEE, 2019.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

Jan Larsen, A Szymkowiak, and Lars Kai Hansen. Probabilistic hierarchical clustering with labeled and unlabeled data. *International Journal of Knowledge Based Intelligent Engineering Systems*, 6 (1):56–63, 2002.

N. Linial, E. London, and Y. Rabinovich. The geometry of graphs and some of its algorithmic applications. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, 1994.

Emile Mathieu and Maximilian Nickel. Riemannian continuous normalizing flows. *Advances in Neural Information Processing Systems*, 2020.

Emile Mathieu, Charline Le Lan, Chris J. Maddison, Ryota Tomioka, and Yee Whye Teh. Continuous hierarchical representations with poincaré variational auto-encoders. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2019.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 2015.

Yoshihiro Nagano, Shoichiro Yamaguchi, Yasuhiro Fujita, and f Masanori Koyama. A wrapped normal distribution on hyperbolic space for gradient-based learning. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, Proceedings of Machine Learning Research. PMLR, 2019.

Kim Anh Nguyen, Maximilian Köper, Sabine Schulte im Walde, and Ngoc Thang Vu. Hierarchical embeddings for hypernymy detection and directionality. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, September 2017.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *International Conference on Machine Learning*, 2011.

Maximillian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2017.

Maximillian Nickel and Douwe Kiela. Learning continuous hierarchies in the Lorentz model of hyperbolic geometry. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, Proceedings of Machine Learning Research. PMLR, 2018.

Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning*. JMLR.org, 2015.

Joel W Robbin and Dietmar A Salamon. *Introduction to differential geometry*. Springer Nature, 2022.

Salem Said, Lionel Bombrun, and Yannick Berthoumieu. New riemannian priors on the univariate normal model. *Entropy*, 2014.

Ruslan Salakhutdinov, Joshua Tenenbaum, and Antonio Torralba. One-shot learning with a hierarchical nonparametric bayesian model. In *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, pages 195–206. JMLR Workshop and Conference Proceedings, 2012.

Rik Sarkar. Low distortion delaunay embedding of trees in hyperbolic plane. In *Proceedings of the 19th International Conference on Graph Drawing*. Springer-Verlag, 2011.

Yuval Shavitt and Tomer Tankel. Hyperbolic embedding of internet graph for distance estimation and overlay construction. *IEEE ACM Transactions on Networking*, 2008.

Zequn Sun, Muhao Chen, Wei Hu, Chengming Wang, Jian Dai, and Wei Zhang. Knowledge association with hyperbolic knowledge graph embeddings. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, November 2020.

Yee Teh, Hal Daume III, and Daniel M Roy. Bayesian agglomerative clustering with coalescents. *Advances in neural information processing systems*, 20, 2007.

Alexandru Tifrea, Gary Becigneul, and Octavian-Eugen Ganea. Poincare glove: Hyperbolic word embeddings. In *International Conference on Learning Representations*, 2019.

Luke Vilnis and Andrew McCallum. Word representations via gaussian embedding. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

Xiaohan Zhao, Alessandra Sala, Haitao Zheng, and Ben Y. Zhao. Efficient shortest paths on massive social graphs. In *7th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom)*, 2011.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 6
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] The code is provided with the supplemental materials.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [No]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [No]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Proof of Proposition 1

We prove the first proposition in Section 3.1 in this section. Through the proposition, we first show that straight lines that pass through the origin in the Euclidean space are transformed into the geodesics in hyperbolic space by Equation 3.

Proposition 1. Suppose $\ell_{\boldsymbol{s}}(t) = t\boldsymbol{s} \in \mathbb{R}^n$ be a line passing through the origin, where $\boldsymbol{s} \in \mathbb{R}^n$ is a directional vector. Then the curve $f_{\boldsymbol{\mu}}(\ell_{\boldsymbol{s}}(t))$ in the Lorentz model $\mathbb{L}^n$ becomes a geodesic.

Proof. Let $\boldsymbol{u} := [x_0(\boldsymbol{u}), x_{1:}(\boldsymbol{u})] \in \mathbb{L}^n$ be given, where $x_0 : \mathbb{L}^n \to \mathbb{R}$ and $x_{1:} : \mathbb{L}^n \to \mathbb{R}^n$ denotes the projections, i.e., $x_i(\boldsymbol{u}) = u_i$. Then,

$$\begin{aligned}
\mathrm{PT}_{\mathbf{0}_{\mathcal{L}} \to \boldsymbol{\mu}}([0, t\boldsymbol{s}]) &= [0, t\boldsymbol{s}] + \frac{1}{x_0(\boldsymbol{\mu}) + 1} \langle \boldsymbol{\mu} - x_0(\boldsymbol{\mu}) \cdot \mathbf{0}_{\mathcal{L}}, [0, t\boldsymbol{s}] \rangle_{\mathcal{L}} (\mathbf{0}_{\mathcal{L}} + \boldsymbol{\mu}) \\
&= [0, t\boldsymbol{s}] + \frac{1}{x_0(\boldsymbol{\mu}) + 1} \langle [0, x_{1:}(\boldsymbol{\mu})], [0, t\boldsymbol{s}] \rangle_{\mathcal{L}} [x_0(\boldsymbol{\mu}) + 1, x_{1:}(\boldsymbol{\mu})] \\
&= [0, t\boldsymbol{s}] + \frac{1}{x_0(\boldsymbol{\mu}) + 1} \langle x_{1:}(\boldsymbol{\mu}), t\boldsymbol{s} \rangle [x_0(\boldsymbol{\mu}) + 1, x_{1:}(\boldsymbol{\mu})] \\
&= t \left[\langle x_{1:}(\boldsymbol{\mu}), \boldsymbol{s} \rangle, \boldsymbol{s} + \langle x_{1:}(\boldsymbol{\mu}), \boldsymbol{s} \rangle \frac{x_{1:}(\boldsymbol{\mu})}{x_0(\boldsymbol{\mu}) + 1} \right].
\end{aligned}$$

Now, let $\boldsymbol{v} := \left[\langle x_{1:}(\boldsymbol{\mu}), \boldsymbol{s} \rangle, \boldsymbol{s} + \langle x_{1:}(\boldsymbol{\mu}), \boldsymbol{s} \rangle \frac{x_{1:}(\boldsymbol{\mu})}{x_0(\boldsymbol{\mu}) + 1} \right]$ and $c := \sqrt{\langle \boldsymbol{v}, \boldsymbol{v} \rangle_{\mathcal{L}}}$. Then,

$$\begin{aligned}
f_{\boldsymbol{\mu}}(l_{\boldsymbol{s}}) &= \exp_{\boldsymbol{\mu}}(\mathrm{PT}_{\mathbf{0}_{\mathcal{L}} \to \boldsymbol{\mu}}((0, t\boldsymbol{s}))) \\
&= \exp_{\boldsymbol{\mu}}(t\boldsymbol{v}) \\
&= \cosh(ct)\boldsymbol{\mu} + \sinh(ct)\frac{\boldsymbol{v}}{c}.
\end{aligned}$$

Recall that the geodesic of the Lorentz model is $\cosh(t)\boldsymbol{x} + \sinh(t)\boldsymbol{y}$ where $\boldsymbol{x} \in \mathbb{L}^n, \langle \boldsymbol{y}, \boldsymbol{y} \rangle_{\mathcal{L}} = 1$ and $\boldsymbol{y} \in \mathcal{T}_{\boldsymbol{x}}\mathbb{L}^n$ (Robbin and Salamon, 2022). □

The proposition indicates that every straight line that passes through the origin including the principal axes, is transformed into a geodesic in hyperbolic space.

B Proof of Proposition 2

We prove the second proposition in Section 3.1 to show the geometrical properties of HWN. Based on the first proposition, we provide our main proposition, which fully characterizes the structure of principal axes when projected to the Poincaré disk model:

Proposition 2. Let $\mathrm{Proj}(\boldsymbol{u})$ be the projection function from Lorentz model to Poincaré model, i.e., $\mathrm{Proj}(\boldsymbol{u}) = \frac{x_{1:}(\boldsymbol{u})}{x_0(\boldsymbol{u})+1}, \forall \boldsymbol{u} \in \mathbb{L}^n$. Let $\ell_{\boldsymbol{s}}$ be a principal axis of the normal distribution defined in $\mathbb{R}^n$ and $\boldsymbol{\mu}$ the mean of HWN in $\mathbb{L}^n$, then $\boldsymbol{s}$ is the tangent vector of $\mathrm{Proj}(f_{\boldsymbol{\mu}}(\ell_{\boldsymbol{s}}))$ on $\mathrm{Proj}(\boldsymbol{\mu})$.

Proof. First, we project the transformed principal axis to the Poincaré disk model as:

$$\begin{aligned}
\mathrm{Proj}\left(f_{\boldsymbol{\mu}}(l_{\boldsymbol{s}})\right) &= \mathrm{Proj}\left(\cosh(ct)\boldsymbol{\mu} + \sinh(ct)\frac{\boldsymbol{v}}{c}\right) \\
&= \mathrm{Proj}\left(\left[\cosh(ct)x_0(\boldsymbol{\mu}) + \sinh(ct)\frac{x_0(\boldsymbol{v})}{c}, \cosh(ct)x_{1:}(\boldsymbol{\mu}) + \sinh(ct)\frac{x_{1:}(\boldsymbol{v})}{c}\right]\right) \\
&= \frac{\cosh(ct)x_{1:}(\boldsymbol{\mu}) + \sinh(ct)x_{1:}(\boldsymbol{v})/c}{\cosh(ct)x_0(\boldsymbol{\mu}) + \sinh(ct)x_0(\boldsymbol{v})/c + 1}.
\end{aligned}$$

Then, the derivative of the projected curve with respect to t is derived as:

$$\begin{aligned}
\frac{\partial}{\partial t}\mathrm{Proj}\left(f_{\boldsymbol{\mu}}(l_{\boldsymbol{s}})\right) &= \frac{\partial}{\partial t}\frac{\cosh(ct)x_{1:}(\boldsymbol{\mu}) + \sinh(ct)x_{1:}(\boldsymbol{v})/c}{\cosh(ct)x_0(\boldsymbol{\mu}) + \sinh(ct)x_0(\boldsymbol{v})/c + 1} \\
&= \frac{\sinh(ct)x_{1:}(\boldsymbol{\mu})c + \cosh(ct)x_{1:}(\boldsymbol{v})}{\cosh(ct)x_0(\boldsymbol{\mu}) + \sinh(ct)x_0(\boldsymbol{v})/c + 1} \\
&\quad - \frac{(\sinh(ct)x_0(\boldsymbol{\mu})c + \cosh(ct)x_0(\boldsymbol{v}))(\cosh(ct)x_{1:}(\boldsymbol{\mu}) + \sinh(ct)x_{1:}(\boldsymbol{v})/c)}{\left(\cosh(ct)x_0(\boldsymbol{\mu}) + \sinh(ct)x_0(\boldsymbol{v})/c + 1\right)^2}.
\end{aligned}$$

As $\mathrm{Proj}(\boldsymbol{\mu})$ is the point of the curve at $t = 0$, the tangent vector of the curve on $\mathrm{Proj}(\boldsymbol{\mu})$ can be computed by substituting $t = 0$:

$$\begin{aligned}
\frac{\partial}{\partial t}\mathrm{Proj}\left(f_{\boldsymbol{\mu}}(l_{\boldsymbol{s}})\right)\Big|_{t=0} &= \frac{x_{1:}(\boldsymbol{v})}{x_0(\boldsymbol{\mu}) + 1} - \frac{x_0(\boldsymbol{v})x_{1:}(\boldsymbol{\mu})}{(x_0(\boldsymbol{\mu}) + 1)^2} \\
&= \frac{\boldsymbol{s}}{x_0(\boldsymbol{\mu}) + 1} + \frac{\langle x_{1:}(\boldsymbol{\mu}), \boldsymbol{s}\rangle x_{1:}(\boldsymbol{\mu})}{(x_0(\boldsymbol{\mu}) + 1)^2} - \frac{\langle x_{1:}(\boldsymbol{\mu}), \boldsymbol{s}\rangle x_{1:}(\boldsymbol{\mu})}{(x_0(\boldsymbol{\mu}) + 1)^2} \\
&= \frac{\boldsymbol{s}}{x_0(\boldsymbol{\mu}) + 1} \\
&\propto \boldsymbol{s}.
\end{aligned}$$

□

The proposition reveals that the principal axes of the HWN are locally parallel to the standard bases in the Poincaré disk model. To visualize the proposition, we plot the contour line and the principal axes of the two-dimensional diagonal covariance normal distribution before and after the transformation in Figure 2. We observe that the tangent lines of the transformed principal axes are parallel to the standard bases in hyperbolic space.

C Details of Rotated Hyperbolic Wrapped Normal Distribution

The details of RoWN, i.e. sampling and the probability density computation, are described in this section. We start with a mean vector $\boldsymbol{\mu} \in \mathbb{L}^n$ and a diagonal covariance matrix Σ as in the standard HWN. Based on the mean vector, we construct RoWN by rotating the covariance matrix as follows:

1. Compute the rotation matrix $\boldsymbol{R}$ that rotates the x-axis ($[\pm 1, \dots, 0] \in \mathbb{R}^n$) to $\boldsymbol{\mu}_{1:}$.
2. Substitute the covariance matrix of Gaussian normal with $\boldsymbol{R}\Sigma\boldsymbol{R}^T$.

Thus, the rotation matrix $\boldsymbol{R}$, which rotates a unit vector from $\boldsymbol{x}$ to $\boldsymbol{y}$, can be computed as:

$$\boldsymbol{R} = \boldsymbol{I} + (\boldsymbol{y}^T\boldsymbol{x} - \boldsymbol{x}^T\boldsymbol{y}) + \frac{1}{1 + \langle \boldsymbol{x}, \boldsymbol{y} \rangle}(\boldsymbol{y}^T\boldsymbol{x} - \boldsymbol{x}^T\boldsymbol{y})^2. \tag{6}$$

Algorithm 2 shows the entire algorithm of constructing RoWN.

Note that the construction is straightforward but still keeps the following benefits of the HWN: 1) The sampling can be done efficiently, and 2) the computation of the probability density of the samples is tractable. As the HWN provides a tractable probability density function for any kind of covariance matrix, we can easily compute the probability density of a given sample from RoWN. For the details of sampling and probability density computation, see Algorithm 3 & 4.

Algorithm 2 RoWN($\boldsymbol{\mu}, \Sigma$)

Input Mean $\boldsymbol{\mu} \in \mathbb{L}^n$, diagonal covariance matrix $\Sigma \in \mathbb{R}^{n \times n}$

Output Rotated covariance matrix $\hat{\Sigma}$.

$\boldsymbol{x} = [\pm 1, \dots, 0] \in \mathbb{R}^n, \boldsymbol{y} = \boldsymbol{\mu}_{1:}/\|\boldsymbol{\mu}_{1:}\|$ ▷ $\pm$ is determined by the sign of $\boldsymbol{\mu}_0$

$\boldsymbol{R} = \boldsymbol{I} + (\boldsymbol{y}^T\boldsymbol{x} - \boldsymbol{x}^T\boldsymbol{y}) + (\boldsymbol{y}^T\boldsymbol{x} - \boldsymbol{x}^T\boldsymbol{y})^2/(1 + \langle \boldsymbol{x}, \boldsymbol{y} \rangle)$

return $\hat{\Sigma} = \boldsymbol{R}\Sigma\boldsymbol{R}^T$

Algorithm 3 Sampling process with the rotated hyperbolic wrapped normal distribution

Input Mean $\boldsymbol{\mu} \in \mathbb{L}^n$, diagonal covariance matrix $\Sigma \in \mathbb{R}^{n \times n}$

Output Sample $\boldsymbol{z} \in \mathbb{L}^n$

Construct $\hat{\Sigma} = \text{RoWN}(\boldsymbol{\mu}, \Sigma)$

Sample $\boldsymbol{v} \sim \mathcal{N}(\boldsymbol{0}, \hat{\Sigma})$

return $\boldsymbol{z} = f_{\boldsymbol{\mu}}(\boldsymbol{v})$

Algorithm 4 Probability density computation of the rotated hyperbolic wrapped normal distribution

Input Mean $\boldsymbol{\mu} \in \mathbb{L}^n$, diagonal covariance matrix $\Sigma \in \mathbb{R}^{n \times n}$, sample $\boldsymbol{z} \in \mathbb{L}^n$

Output Log probability of $\boldsymbol{z}$.

Construct $\hat{\Sigma} = \text{RoWN}(\boldsymbol{\mu}, \Sigma)$

$\boldsymbol{u} = \log_{\boldsymbol{\mu}}(\boldsymbol{z})$

$\boldsymbol{v} = \text{PT}_{\boldsymbol{\mu} \to \boldsymbol{0}_{\mathcal{L}}}(\boldsymbol{u})$ ▷ $\boldsymbol{v} = f_{\boldsymbol{\mu}}^{-1}(\boldsymbol{z})$

return (log probability of $\boldsymbol{v}_{1:}$ from $\mathcal{N}(\boldsymbol{0}, \hat{\Sigma})$) $- (n-1)(\log \sinh \|\boldsymbol{u}\|_{\mathcal{L}} - \log \|\boldsymbol{u}\|_{\mathcal{L}})$

D Experimental Details

In this section, we provide the details of the experiments in Section 4.

D.1 Baselines

For all the experiments, we compare the performance of RoWN with four baselines: the normal distribution in the Euclidean space, the isotropic HWN (Nagano et al., 2019), the diagonal HWN, and the full covariance HWN.

In the process of constructing the distributions for each application, i.e. the variational distribution of VAE and the embedding distribution of probabilistic word embedding, the distributions except the full covariance HWN have a diagonal matrix as an input, and then the softplus operation is used to make it positive. The full covariance HWN has a 2D matrix $\Sigma \in \mathbb{R}^{n \times n}$ as an input and constructs a covariance matrix as $\Sigma\Sigma^T + \epsilon \boldsymbol{I}$, to match the positive-definite property, where ϵ is set to $1\mathrm{e}{-9}$ in our experiments. For the mean value, we concatenate zero at the first dimension and transport it to the Lorentz model using $\exp_{\mathbf{0}_{\mathcal{L}}}$.

For the hyperbolic VAE models, we use $\log_{\mathbf{0}_{\mathcal{L}}}$ to transform the input of the decoder to the Euclidean space, as suggested in Mathieu et al. (2019).

D.2 Noisy synthetic binary tree

Experimental setting. A synthetic binary tree dataset is first used to show the performance of representing hierarchy in Nagano et al. (2019), where each node in a tree corresponds to a sequence of binary values. Figure 4a shows an example of the depth three binary tree, where a parent and child only differ in one digit. We add spherical noises to the nodes in the same level of hierarchy as described in Figure 4a as the noisy samples. With the data points with additional noises, we can create a dataset containing a local-level variation in the hierarchy. For the experiments, we uniformly sample the spherical noise from $[0, \pi/4]$.

We train hyperbolic VAE on *noisy synthetic binary tree* with varying depth from 4 to 7. For each depth d, we use a three-layer fully connected neural network as the architecture where the number of hidden units is 2^{d+3} and the latent dimension is d. We use ReLU as the activation function for each layer except the last layer of the encoder and decoder. The overall architecture is shown in Table 5 and Table 6. We use a Gaussian negative log-likelihood loss for the reconstruction loss with fixed $\sigma = 0.01$, which is selected for sufficient reconstruction performance on the train set.

Table 5: Encoder architecture for *noisy synthetic binary tree*

Layer	Output dim	Activation
FC	2^{d+3}	ReLU
FC	2^{d+3}	ReLU
FC	$2d$	None

Table 6: Decoder architecture for *noisy synthetic binary tree*

Layer	Output dim	Activation
FC	2^{d+3}	ReLU
FC	2^{d+3}	ReLU
FC	$2^d - 1$	None

Results. We report 1) the correlation between the hamming distance and the embedding distance and 2) the correlation between the depth and the norm of the embeddings. The first correlation is computed over all possible pairs of test points. For the norm of the hyperbolic embeddings, we use the Poincaré norm, which can be calculated by projecting the Lorentz model embedding to the Poincaré disk model.

As Table 7 shows, while all the models show similar performance with respect to the ELBO, the full covariance HWN and RoWN show better performance than the diagonal HWN except depth six, outperforming the Euclidean model in every setting. RoWN preserves the depth information better than the other distributions in general. We additionally visualize the variational mean obtained by training the tree of depth three in Figure 4b, where the hierarchical structure is well preserved in the hyperbolic embedding space.

Table 7: Results of *noisy synthetic binary tree*. The results are averaged over 10 runs. The hyperbolic models outperform the Euclidean model in all settings. Overall, RoWN preserves the hierarchical information better than the other distributions.

		depth			
		4	5	6	7
Correlation w/ distance	Euclidean	$0.748_{\pm.032}$	$0.740_{\pm.013}$	$0.741_{\pm.008}$	$0.733_{\pm.014}$
	HWN (isotropic Σ)	$0.773_{\pm.030}$	$0.809_{\pm.016}$	$0.798_{\pm.008}$	$0.735_{\pm.022}$
	HWN (diagonal Σ)	$0.814_{\pm.008}$	$0.791_{\pm.023}$	$0.817_{\pm.010}$	$0.759_{\pm.025}$
	HWN (full Σ)	$0.827_{\pm.015}$	$0.798_{\pm.026}$	$0.798_{\pm.010}$	$0.794_{\pm.014}$
	RoWN	$0.820_{\pm.015}$	$0.807_{\pm.017}$	$0.822_{\pm.017}$	$0.788_{\pm.016}$
Correlation w/ depth	Euclidean	$0.762_{\pm.117}$	$0.807_{\pm.038}$	$0.712_{\pm.054}$	$0.612_{\pm.049}$
	HWN (isotropic Σ)	$0.902_{\pm.033}$	$0.867_{\pm.034}$	$0.811_{\pm.029}$	$0.602_{\pm.066}$
	HWN (diagonal Σ)	$0.918_{\pm.028}$	$0.808_{\pm.076}$	$0.862_{\pm.035}$	$0.697_{\pm.076}$
	HWN (full Σ)	$0.956_{\pm.015}$	$0.878_{\pm.051}$	$0.870_{\pm.033}$	$0.815_{\pm.055}$
	RoWN	$0.930_{\pm.026}$	$0.911_{\pm.027}$	$0.901_{\pm.034}$	$0.827_{\pm.047}$
Test ELBO	Euclidean	$22.591_{\pm.183}$	$55.168_{\pm.092}$	$124.374_{\pm.093}$	$266.854_{\pm.199}$
	HWN (isotropic Σ)	$22.026_{\pm.201}$	$54.054_{\pm.158}$	$122.981_{\pm.145}$	$265.316_{\pm.217}$
	HWN (diagonal Σ)	$22.480_{\pm.144}$	$54.540_{\pm.117}$	$123.444_{\pm.110}$	$265.704_{\pm.154}$
	HWN (full Σ)	$22.371_{\pm.136}$	$55.032_{\pm.141}$	$124.125_{\pm.189}$	$266.499_{\pm.112}$
	RoWN	$22.354_{\pm.138}$	$54.648_{\pm.142}$	$123.606_{\pm.066}$	$266.146_{\pm.112}$

Decomposition of the radial and angular dependency. To show how well the angular and radial representations are decomposed, we compute the Pearson correlation between the radial axis and the angular axis, which has been shown to be an effective measure of variable dependency in disentangled representation learning (Jo and Seo, 2019; Horan et al., 2021). Table 8 shows that the absolute value of the correlation is lower or similar to 0.1 in all the models, including RoWN.

Learnable rotation. We add experiments for the models that learn the rotation direction, which is originally fixed to the direction of $\boldsymbol{\mu}$ (in Algorithm 1, the $\boldsymbol{y}$ vector). Given data, the encoder gives the rotation direction. We test the models on the noisy synthetic binary tree setting in our paper. As shown Table 9, as the depth becomes deeper, the learnable rotation models usually underperform RoWN. Figure 6 shows behaviors of learned representation by an alternative Learnable Rotation 1. Most of the rotation directions are pointing or orthogonal (black line segments on each node) to the direction of its parent node. This implies that the alternatives of RoWN can learn representations of nodes to align not to the root node but to the parent node. We note that these alternatives work well with shallow depths but not with great depths.

D.3 WordNet

Experimental setting. We train a probabilistic word embedding model with WordNet dataset (Fellbaum, 1998). We initialize the mean and variance parameters with $\mathcal{N}(\mathbf{0}, 0.01)$. For the full covariance model, we use a learning rate 0.01. For the other models, we set the learning rate 0.015 for the first 100 epochs and then set the learning rate to 0.6 for the remaining steps as done in (Nickel and Kiela, 2017; Nagano et al., 2019). We evaluate the learned representations by computing the average rank of all the hypernymies. The rank of a given pair of words s and t is computed among the distances between all possible pairs of the words s and t' without hypernymy.

Results. We evaluate the learned representations by computing the average rank of all the hypernymies. The rank of a given pair of words s and t is computed among the distances between all possible pairs of the words s and t' without hypernymy. Table 2 shows the empirical performances of representing the word data. We report the performance with the mean rank (MR) and the mean average precision (mAP). RoWN preserves the hierarchical structure better than the other distributions, while the full covariance HWN fails due to unstable optimization.

Table 8: Correlation between the radian axis and the angular axis.

		depth			
		4	5	6	7
Correlation btw. r and θ_1	Euclidean	$0.144_{\pm.170}$	$0.007_{\pm.105}$	$0.039_{\pm.106}$	$-0.026_{\pm.093}$
	HWN (diagonal Σ)	$0.025_{\pm.150}$	$0.015_{\pm.137}$	$0.110_{\pm.065}$	$-0.003_{\pm.134}$
	HWN (full Σ)	$-0.053_{\pm.216}$	$0.049_{\pm.136}$	$-0.017_{\pm.169}$	$0.012_{\pm.066}$
	RoWN	$0.080_{\pm.163}$	$0.030_{\pm.097}$	$0.112_{\pm.093}$	$0.025_{\pm.083}$
Correlation btw. r and θ_2	Euclidean	$0.116_{\pm.232}$	$-0.039_{\pm.210}$	$0.109_{\pm.131}$	$0.006_{\pm.095}$
	HWN (diagonal Σ)	$0.066_{\pm.170}$	$-0.021_{\pm.190}$	$0.044_{\pm.122}$	$-0.024_{\pm.115}$
	HWN (full Σ)	$0.297_{\pm.116}$	$0.113_{\pm.109}$	$0.013_{\pm.122}$	$0.061_{\pm.103}$
	RoWN	$0.013_{\pm.178}$	$0.025_{\pm.173}$	$-0.004_{\pm.115}$	$0.064_{\pm.082}$
Correlation btw. r and θ_3	Euclidean	$0.067_{\pm.220}$	$-0.110_{\pm.220}$	$-0.016_{\pm.159}$	$0.011_{\pm.098}$
	HWN (diagonal Σ)	$0.123_{\pm.252}$	$-0.139_{\pm.123}$	$-0.019_{\pm.133}$	$-0.095_{\pm.101}$
	HWN (full Σ)	$-0.053_{\pm.144}$	$-0.088_{\pm.107}$	$0.106_{\pm.120}$	$0.024_{\pm.079}$
	RoWN	$0.127_{\pm.253}$	$-0.120_{\pm.169}$	$-0.012_{\pm.120}$	$-0.012_{\pm.083}$
Correlation btw. r and θ_4	Euclidean	-	$-0.015_{\pm.073}$	$0.013_{\pm.081}$	$0.026_{\pm.065}$
	HWN (diagonal Σ)	-	$-0.047_{\pm.117}$	$-0.035_{\pm.147}$	$0.080_{\pm.096}$
	HWN (full Σ)	-	$0.079_{\pm.150}$	$0.042_{\pm.108}$	$0.086_{\pm.102}$
	RoWN	-	$-0.031_{\pm.116}$	$-0.070_{\pm.112}$	$0.086_{\pm.115}$
Correlation btw. r and θ_5	Euclidean	-	-	$0.082_{\pm.075}$	$-0.029_{\pm.109}$
	HWN (diagonal Σ)	-	-	$0.022_{\pm.101}$	$-0.041_{\pm.097}$
	HWN (full Σ)	-	-	$-0.058_{\pm.111}$	$0.076_{\pm.064}$
	RoWN	-	-	$-0.016_{\pm.111}$	$-0.030_{\pm.112}$
Correlation btw. r and θ_6	Euclidean	-	-	-	$-0.018_{\pm.073}$
	HWN (diagonal Σ)	-	-	-	$-0.030_{\pm.082}$
	HWN (full Σ)	-	-	-	$0.035_{\pm.103}$
	RoWN	-	-	-	$0.006_{\pm.067}$

Optimization issue in training full covariance HWN. In the results, we find that the full covariance HWN shows poor performance compared to the other models. We run extensive experiments to show that the full covariance HWN is difficult to optimize especially in WordNet. Figure 7 shows the performance of the full covariance HWN with varying hyperparameters, i.e. learning rate, the burn-in factor, and the initialization method, which seems to be poor whatever we choose. We conducted

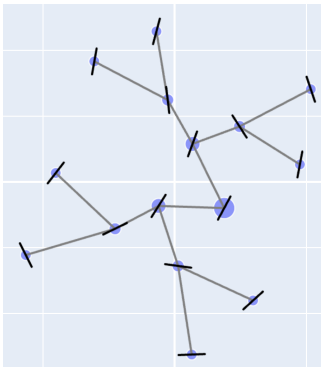


Figure 6: Visualization of the representations from learnable rotation model. The representations are from the Learnable Rotation 1 model learned on the depth 4 noisy synthetic binary tree with latent dimension 2. The size of the circles denotes the depth, where the biggest circle denotes the root node. Most of the rotation directions are pointing or orthogonal (black line segments on each node) to the direction of its parent node.

Table 9: Results of learnable rotation models. Learnable Rotation 1 model outputs the rotation direction parallel to the mean and variance, while Learnable Rotation 2 model outputs the rotation direction by feeding the mean to a fully connected layer.

		depth			
		4	5	6	7
Correlation w/ distance	RoWN	$0.820_{\pm.015}$	$0.807_{\pm.017}$	$0.822_{\pm.017}$	$0.788_{\pm.016}$
	Learnable Rotation 1	$0.820_{\pm.013}$	$0.804_{\pm.016}$	$0.808_{\pm.013}$	$0.770_{\pm.028}$
	Learnable Rotation 2	$0.817_{\pm.018}$	$0.811_{\pm.018}$	$0.801_{\pm.008}$	$0.772_{\pm.021}$
Correlation w/ depth	RoWN	$0.930_{\pm.026}$	$0.911_{\pm.027}$	$0.901_{\pm.034}$	$0.827_{\pm.047}$
	Learnable Rotation 1	$0.952_{\pm.018}$	$0.903_{\pm.037}$	$0.845_{\pm.057}$	$0.711_{\pm.072}$
	Learnable Rotation 2	$0.948_{\pm.015}$	$0.907_{\pm.019}$	$0.844_{\pm.013}$	$0.724_{\pm.058}$
Test ELBO	RoWN	$22.354_{\pm.138}$	$54.648_{\pm.142}$	$123.606_{\pm.066}$	$266.146_{\pm.112}$
	Learnable Rotation 1	$22.342_{\pm.097}$	$54.741_{\pm.111}$	$123.619_{\pm.180}$	$266.076_{\pm.159}$
	Learnable Rotation 2	$22.286_{\pm.120}$	$54.529_{\pm.067}$	$123.382_{\pm.161}$	$265.962_{\pm.217}$

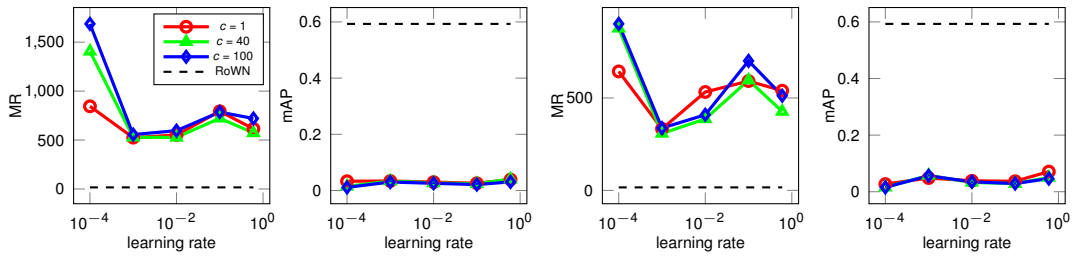
an additional analysis on what causes the optimization instability and found that the number of samples used to approximate the KL divergence is critical to full covariance HWN, especially in the Wordnet dataset. In VAE, we can observe more stable results. We speculate that the stability improved since the relatively simple prior (standard normal distribution) is employed with the full covariance variational distribution. Table 11 shows that as the number of training samples increases, the performance increases, but the result is still poor, and using more training samples leads to an additional computation time.

D.4 Atari 2600 Breakout

Experimental setting. To learn the implicit hierarchy that can be observed from the trajectories of Breakout, we train the VAE models with the Atari 2600 Breakout images. The images of Breakout are collected by using a pre-trained Deep Q-network (Mnih et al., 2015) and divided into a training set and test set with 90,000 and 10,000 images respectively. We label each image with the score obtained from the game environment. So the labels are correlated to the number of broken blocks. To train VAE, we use a DCGAN-based architecture, which was originally used to evaluate HWN in Nagano et al. (2019). The detailed architecture is provided in Table 12 and Table 13. We use binary cross-entropy loss for the reconstruction loss.

Table 10: Results of *WordNet*. The results are an average of 5 runs. Based on the rank of hypernymy pairs among non-hypernymy pairs, we report the mean rank (MR) and mean average precision (mAP) for evaluation.

		latent dimension		
		5	10	20
MR	Euclidean	$13.968_{\pm0.504}$	$3.862_{\pm0.281}$	$1.955_{\pm0.157}$
	HWN (isotropic Σ)	$14.568_{\pm2.203}$	$4.470_{\pm0.669}$	$3.125_{\pm0.455}$
	HWN (diagonal Σ)	$16.590_{\pm1.146}$	$3.891_{\pm0.447}$	$2.062_{\pm0.088}$
	HWN (full Σ)	$557.309_{\pm18.006}$	$466.513_{\pm75.142}$	$599.140_{\pm18.916}$
	RoWN	$16.271_{\pm2.985}$	$2.888_{\pm0.162}$	$1.783_{\pm0.090}$
mAP	Euclidean	$0.565_{\pm0.014}$	$0.801_{\pm0.020}$	$0.902_{\pm0.008}$
	HWN (isotropic Σ)	$0.617_{\pm0.012}$	$0.820_{\pm0.013}$	$0.847_{\pm0.017}$
	HWN (diagonal Σ)	$0.565_{\pm0.020}$	$0.805_{\pm0.015}$	$0.905_{\pm0.007}$
	HWN (full Σ)	$0.032_{\pm0.003}$	$0.063_{\pm0.005}$	$0.079_{\pm0.021}$
	RoWN	$0.593_{\pm0.024}$	$0.844_{\pm0.009}$	$0.921_{\pm0.005}$



(a) MR of full initialization (b) mAP of full initialization (c) MR of diagonal initialization (d) mAP of diagonal initialization

Figure 7: Varying hyperparameters of the full covariance HWN on WordNet. We run several hyperparameters combination of full covariance HWN on WordNet. We fix the latent dimension to 5 and burn-in epochs to 100. We vary the learning rate from $1e{-}4$ to 0.6 and the factor c, which is used to reduce the learning rate in the burn-in steps, from 1 to 100. We run 10,000 epochs. (a,b) When we initialize the entire covariance matrix with $\mathcal{N}(0, 0.01)$, the full covariance HWN show poor performance with any hyper-parameter combinations. (c,d) Initializing only the diagonal entries with $\mathcal{N}(0, 0.01)$ and let the remaining part to zero improves the performance of the full covariance HWN but it still performs worse than RoWN.

Table 11: Varying samples of the full covariance HWN on WordNet.

# of samples	1	50	100
MR	$326.185_{\pm 7.610}$	$73.572_{\pm 9.926}$	$69.964_{\pm 10.457}$
mAP	$0.056_{\pm 0.004}$	$0.241_{\pm 0.021}$	$0.241_{\pm 0.019}$
Runtime (s/epoch)	6.288	6.748	8.163

Results. To evaluate the models, we measure the correlation between the norm of the test images and the labeled scores. For the norm of the hyperbolic embeddings, we use the Poincaré norm, which can be calculated by projecting the Lorentz model embedding to the Poincaré disk model. The results are reported in Table 14. While all the models show similar representation power with respect to ELBO, RoWN and the full covariance HWN outperform the diagonal HWN. Especially, RoWN aligns the hierarchical structures better with respect to the norm in low latent dimensions.

Nagano et al. (2019) report a higher score of correlation in latent dimension 20, but we find some issues with the result. First, Nagano et al. (2019) compute the correlation between the labeled scores and the norm in the tangent space ($\boldsymbol{v}$ vector from Algorithm 4), not the Poincaré norm. The projection function in Proposition 2 depends on the first element of the input vector. Thus the Poincaré norm is not proportional to the $\boldsymbol{v}$ norm, and computing the correlation with the $\boldsymbol{v}$ norm will show different behavior compared to the correlation with the Poincaré norm. Second, the reproduction results obtained from the code by the official repository[1] are far from the reported correlation. Our reproduction results of the correlation between the labeled scores and the $\boldsymbol{v}$ norm show 0.616, and between the Poincaré norm show 0.501 averaged over four runs.

Qualitative results. Figure 8 shows more examples of generated images from VAE models trained on Breakout images with two dimensional latent space.

[1] `https://github.com/pfnet-research/hyperbolic_wrapped_distribution`

Table 12: Encoder architecture for Breakout

Layer	Output dim	Activation
Conv2d	$80 \times 80 \times 16$	ReLU
Conv2d	$40 \times 40 \times 32$	ReLU
Conv2d	$40 \times 40 \times 32$	ReLU
Conv2d	$20 \times 20 \times 64$	ReLU
Conv2d	$20 \times 20 \times 64$	ReLU
Conv2d	$10 \times 10 \times 64$	ReLU
FC	$2 \times$ latent dimension	None

Table 13: Decoder architecture for Breakout

Layer	Output dim	Activation
FC	$10 \times 10 \times 64$	ReLU
ConvTranspose2d	$20 \times 20 \times 32$	ReLU
Conv2d	$20 \times 20 \times 32$	ReLU
ConvTranspose2d	$40 \times 40 \times 16$	ReLU
Conv2d	$40 \times 40 \times 16$	ReLU
ConvTranspose2d	$80 \times 80 \times 1$	Sigmoid

Table 14: Results of *Atari 2600 Breakout*. The results are averaged over 10 runs. We measure the correlation between the score of an image and the Poincaré norm of the variational mean.

		latent dimension		
		10	15	20
Correlation btw. score and norm	Euclidean	$0.379_{\pm.007}$	$0.436_{\pm.029}$	$0.479_{\pm.020}$
	HWN (isotropic Σ)	$0.513_{\pm.012}$	$0.598_{\pm.021}$	$0.607_{\pm.015}$
	HWN (diagonal Σ)	$0.478_{\pm.011}$	$0.513_{\pm.006}$	$0.513_{\pm.008}$
	HWN (full Σ)	$0.483_{\pm.011}$	$0.520_{\pm.009}$	$0.563_{\pm.010}$
	RoWN	$0.497_{\pm.014}$	$0.556_{\pm.014}$	$0.561_{\pm.029}$
Test ELBO	Euclidean	$-1269.044_{\pm.241}$	$-1269.624_{\pm.258}$	$-1269.682_{\pm.178}$
	HWN (isotropic Σ)	$-1271.018_{\pm.440}$	$-1272.139_{\pm.170}$	$-1272.914_{\pm.118}$
	HWN (diagonal Σ)	$-1269.816_{\pm.272}$	$-1270.725_{\pm.260}$	$-1271.087_{\pm.234}$
	HWN (full Σ)	$-1269.021_{\pm.320}$	$-1269.569_{\pm.206}$	$-1269.882_{\pm.438}$
	RoWN	$-1269.531_{\pm.212}$	$-1270.203_{\pm.211}$	$-1270.967_{\pm.183}$

E Discussion on Root Placement

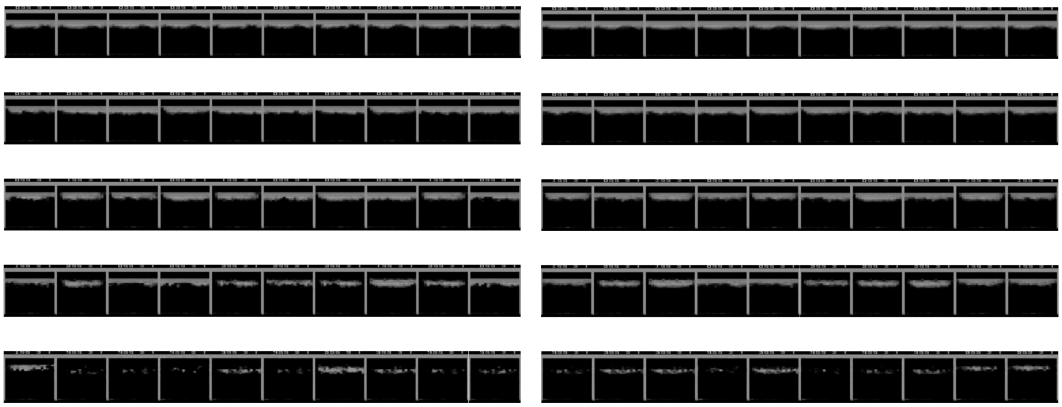
Note that due to the isometry of the geometry, the root node can be placed anywhere in hyperbolic space. In other words, infinitely many sets of embeddings preserve the same pairwise distances between the nodes. This reveals that finding the appropriate isometry, where the root node is placed near the origin, is important for using RoWN as the distribution. In this section, we discuss the techniques we used to place the root node near the origin of each application.

E.1 Probabilistic word embedding model

In our experiments, to place the root node near the origin, we have initialized embeddings from $\mathcal{N}(0, 0.01I)$, which are then moved to the Lorentz model using the exponential map, with learning rate warm-up Nagano et al. (2019). The results with different initialization method are shown in Table 3.

E.2 Hyperbolic VAE

When RoWN is used as a variational distribution in the hyperbolic VAE, the application of RoWN is different from the probabilistic word embedding model since the KL divergence encourages all variational means to be close to the prior mean. Suppose we only focus on nodes at a certain depth in a tree. In that case, it can be easily identified that it would be beneficial to have all nodes at the same level of the norm to minimize the geometric mean between the prior and posterior means. Here, we assume that each pair of nodes requires to have a certain amount of distance to minimize the reconstruction error. As shown in Figure 3a, the original HWN is difficult to have the nodes at the same depth with similar norms since the local variation cannot be modeled through the radial direction. Eventually, the root node slightly deviates from the prior mean. With RoWN, as shown in



(a) Diagonal HWN (b) RoWN

Figure 8: Generation results of the VAE models trained on Breakout images. The models are trained on Breakout images with two dimensional latent space. We generate the images from randomly sampled latent vectors having Poincaré norm as 0.1, 0.3, 0.5, 0.7, 0.9.

Figure 3c, all the nodes at the same depth can be placed at a similar norm while preserving their local variations. If this is indeed the case, the root node is likely to be placed near the prior mean since the nodes with different depths will be placed at different levels of norms in the space.