Appendix: A Wrapped Normal Distribution on Hyperbolic Space for Gradient-Based Learning

A. Derivations

A.1. Inverse Exponential Map

As we mentioned in the main text, the exponential map from $T_{\boldsymbol{\mu}}\mathbb{H}^n$ to $\mathbb{H}^n$ is given by

$$\boldsymbol{z} = \exp_{\boldsymbol{\mu}}(\boldsymbol{u}) = \cosh\left(\|\boldsymbol{u}\|_{\mathcal{L}}\right)\boldsymbol{\mu} + \sinh\left(\|\boldsymbol{u}\|_{\mathcal{L}}\right)\frac{\boldsymbol{u}}{\|\boldsymbol{u}\|_{\mathcal{L}}}. \tag{1}$$

Solving (1) for $\boldsymbol{u}$, we obtain

$$\boldsymbol{u} = \frac{\|\boldsymbol{u}\|_{\mathcal{L}}}{\sinh(\|\boldsymbol{u}\|_{\mathcal{L}})}(\boldsymbol{z} - \cosh(\|\boldsymbol{u}\|_{\mathcal{L}})\boldsymbol{\mu}).$$

We still need to obtain the evaluatable expression for $\|\boldsymbol{u}\|_{\mathcal{L}}$. Using the characterization of the tangent space (main text, (2)), we see that

$$\begin{aligned}
\langle \boldsymbol{\mu}, \boldsymbol{u}\rangle_{\mathcal{L}} =& \frac{\|\boldsymbol{u}\|_{\mathcal{L}}}{\sinh(\|\boldsymbol{u}\|_{\mathcal{L}})}\Big(\langle \boldsymbol{\mu}, \boldsymbol{z}\rangle_{\mathcal{L}} - \cosh(\|\boldsymbol{u}\|_{\mathcal{L}})\langle \boldsymbol{\mu}, \boldsymbol{\mu}\rangle_{\mathcal{L}}\Big) = 0,\\
\cosh(\|\boldsymbol{u}\|_{\mathcal{L}}) =& -\langle \boldsymbol{\mu}, \boldsymbol{z}\rangle_{\mathcal{L}},\\
\|\boldsymbol{u}\|_{\mathcal{L}} =& \operatorname{arccosh}(-\langle \boldsymbol{\mu}, \boldsymbol{z}\rangle_{\mathcal{L}}).
\end{aligned}$$

Now, defining $\alpha = -\langle \boldsymbol{\mu}, \boldsymbol{z}\rangle_{\mathcal{L}}$, we can obtain the inverse exponential function as

$$\boldsymbol{u} = \exp^{-1}_{\boldsymbol{\mu}}(\boldsymbol{z}) = \frac{\operatorname{arccosh}(\alpha)}{\sqrt{\alpha^2 - 1}}(\boldsymbol{z} - \alpha\boldsymbol{\mu}). \tag{2}$$

A.2. Inverse Parallel Transport

The parallel transportation on the Lorentz model along the geodesic from $\boldsymbol{\nu}$ to $\boldsymbol{\mu}$ is given by

$$\begin{aligned}
\mathrm{PT}_{\boldsymbol{\nu}\to\boldsymbol{\mu}}(\boldsymbol{v}) &= \boldsymbol{v} - \frac{\langle \exp^{-1}_{\boldsymbol{\nu}}(\boldsymbol{\mu}), \boldsymbol{v}\rangle_{\mathcal{L}}}{d_\ell(\boldsymbol{\nu}, \boldsymbol{\mu})^2}\left(\exp^{-1}_{\boldsymbol{\nu}}(\boldsymbol{\mu}) + \exp^{-1}_{\boldsymbol{\mu}}(\boldsymbol{\nu})\right)\\
&= \boldsymbol{v} + \frac{\langle \boldsymbol{\mu} - \alpha\boldsymbol{\nu}, \boldsymbol{v}\rangle_{\mathcal{L}}}{\alpha + 1}(\boldsymbol{\nu} + \boldsymbol{\mu}),
\end{aligned} \tag{3}$$

where $\alpha = -\langle \boldsymbol{\nu}, \boldsymbol{\mu}\rangle_{\mathcal{L}}$. Next, likewise, for the exponential map, we need to be able to compute the inverse of the parallel transform. Solving (3) for $\boldsymbol{v}$, we get

$$\boldsymbol{v} = \boldsymbol{u} - \frac{\langle \boldsymbol{\mu} - \alpha\boldsymbol{\nu}, \boldsymbol{v}\rangle_{\mathcal{L}}}{\alpha + 1}(\boldsymbol{\nu} + \boldsymbol{\mu}).$$

Now, observing that

$$\begin{aligned}
\langle \boldsymbol{\nu} - \alpha\boldsymbol{\mu}, \boldsymbol{u}\rangle_{\mathcal{L}} =& \langle \boldsymbol{\nu}, \boldsymbol{v}\rangle_{\mathcal{L}} + \frac{\langle \boldsymbol{\mu} - \alpha\boldsymbol{\nu}, \boldsymbol{v}\rangle_{\mathcal{L}}}{\alpha + 1}(\langle \boldsymbol{\nu}, \boldsymbol{\nu}\rangle_{\mathcal{L}} + \langle \boldsymbol{\mu}, \boldsymbol{\nu}\rangle_{\mathcal{L}})\\
=& -\langle \boldsymbol{\mu}, \boldsymbol{v}\rangle_{\mathcal{L}} = -\langle \boldsymbol{\mu} - \alpha\boldsymbol{\nu}, \boldsymbol{v}\rangle_{\mathcal{L}},
\end{aligned}$$

we can write the inverse parallel transport as

$$\boldsymbol{v} = \mathrm{PT}^{-1}_{\boldsymbol{\nu}\to\boldsymbol{\mu}}(\boldsymbol{u}) = \boldsymbol{u} + \frac{\langle \boldsymbol{\nu} - \alpha\boldsymbol{\mu}, \boldsymbol{u}\rangle_{\mathcal{L}}}{\alpha+1}(\boldsymbol{\nu}+\boldsymbol{\mu}). \tag{4}$$

The inverse of parallel transport from $\boldsymbol{\nu}$ to $\boldsymbol{\mu}$ coincides with the parallel transport from $\boldsymbol{\mu}$ to $\boldsymbol{\nu}$.

A.3. Determinant of exponential map

We provide the details of the computation of the log determinant of $\exp_\mu$. Let $\boldsymbol{\mu} \in \mathbb{H}^n$, let $\boldsymbol{u} \in T_{\boldsymbol{\mu}}(\mathbb{H}^n)$, and let $\boldsymbol{v} = \exp_{\boldsymbol{\mu}}(\boldsymbol{u})$. Then the derivative is a map from the tangent space of $T_{\boldsymbol{\mu}}(\mathbb{H}^n)$ at $\boldsymbol{u}$ to the tangent space of $\mathbb{H}^n$ at $\boldsymbol{v}$. The determinant of this derivative will not change by any orthogonal change of basis. Let us choose an orthonormal basis of $T_{\boldsymbol{u}}(T_{\boldsymbol{\mu}}(\mathbb{H}^n)) \cong T_{\boldsymbol{\mu}}(\mathbb{H}^n)$ containing $\bar{\boldsymbol{u}} = \boldsymbol{u}/\|\boldsymbol{u}\|_{\mathcal{L}}$:

$$\{\bar{\boldsymbol{u}}, \boldsymbol{u}'_1, \boldsymbol{u}'_2, ..., \boldsymbol{u}'_{n-1}\}$$

The desired determinant can be computed by tracking how much each element of this basis grows in magnitude under the transformation.

The derivative in the direction of each basis element can be computed as follows:

$$\begin{aligned}
\mathrm{d}\exp_{\boldsymbol{\mu}}(\bar{\boldsymbol{u}}) =& \frac{\partial}{\partial\epsilon}\bigg|_{\epsilon=0} \exp_{\boldsymbol{\mu}}(\boldsymbol{u} + \epsilon\bar{\boldsymbol{u}}) \\
=& \frac{\partial}{\partial\epsilon}\bigg|_{\epsilon=0} \left[\cosh(r+\epsilon)\boldsymbol{\mu} + \sinh(r+\epsilon)\frac{\boldsymbol{u}+\epsilon\bar{\boldsymbol{u}}}{\|\boldsymbol{u}+\epsilon\bar{\boldsymbol{u}}\|_{\mathcal{L}}}\right] \\
=& \sinh(r)\boldsymbol{\mu} + \cosh(r)\bar{\boldsymbol{u}}.
\end{aligned} \tag{5}$$

$$\begin{aligned}
\mathrm{d}\exp_{\boldsymbol{\mu}}(\boldsymbol{u}'_k) =& \frac{\partial}{\partial\epsilon}\bigg|_{\epsilon=0} \exp_{\boldsymbol{\mu}}(\boldsymbol{u} + \epsilon\boldsymbol{u}'_k) \\
=& \frac{\partial}{\partial\epsilon}\bigg|_{\epsilon=0} \left[\cosh(r)\boldsymbol{\mu} + \sinh(r)\frac{\boldsymbol{u}+\epsilon\boldsymbol{u}'}{r}\right] \\
=& \frac{\sinh r}{r}\boldsymbol{u}'.
\end{aligned} \tag{6}$$

In the second line of the computation of the directional derivative with respect to $\boldsymbol{u}'_k$, we used the fact that $\|\boldsymbol{u} + \epsilon\boldsymbol{u}'_k\|_{\mathcal{L}} = \sqrt{\langle\boldsymbol{u},\boldsymbol{u}\rangle_{\mathcal{L}} + \epsilon\langle\boldsymbol{u},\boldsymbol{u}'_k\rangle_{\mathcal{L}} + \epsilon^2\langle\boldsymbol{u}'_k,\boldsymbol{u}'_k\rangle_{\mathcal{L}}} = \|\boldsymbol{u}\|_{\mathcal{L}} + \mathcal{O}(\epsilon^2)$ and that $O(\epsilon^2)$ in the above expression will disappear in the $\epsilon \to 0$ limit of the finite difference. All together, the derivatives computed with respect to our choice of the basis elements are given by

$$\left(\sinh(r)\boldsymbol{\mu} + \cosh(r)\bar{\boldsymbol{u}}, \frac{\sinh r}{r}\boldsymbol{u}'_1, \frac{\sinh r}{r}\boldsymbol{u}'_2, \cdots, \frac{\sinh r}{r}\boldsymbol{u}'_{n-1}\right)$$

The desired determinant is the product of the Lorentzian norms of the vectors of the set above. Because all elements of $T_{\boldsymbol{\mu}}(\mathbb{H}^n)$ are orthogonal with respect to the Lorentzian inner product and because $\|\sinh(r)\boldsymbol{\mu} + \cosh(r)\bar{\boldsymbol{u}}\|_{\mathcal{L}} = 1$ and $\|\sinh(r)/r \cdot \boldsymbol{u}'\|_{\mathcal{L}} = \sinh(r)/r$, we get

$$\det\left(\frac{\partial\exp_{\boldsymbol{\mu}}(\boldsymbol{u})}{\partial\boldsymbol{u}}\right) = \left(\frac{\sinh r}{r}\right)^{n-1}. \tag{7}$$

A.4. Determinant of parallel transport

Next, let us compute the determinant of the parallel transport. Let $\boldsymbol{v} \in T_{\boldsymbol{\mu}_0}\mathbb{H}^n$, and let $\boldsymbol{u} = \mathrm{PT}_{\boldsymbol{\mu}_0\to\boldsymbol{\mu}}(\boldsymbol{v}) \in T_{\boldsymbol{\mu}}\mathbb{H}^n$. The derivative of this map is a map from $T_{\boldsymbol{v}}(T_{\boldsymbol{\mu}_0}(\mathbb{H}^n))$ to $T_{\boldsymbol{u}}(\mathbb{H}^n)$. Let us choose an orthonormal basis $\boldsymbol{\xi}_k$ (In Lorentzian sense). Likewise above, we can compute the desired determinant by tracking how much each element of this basis grows in magnitude under the transformation.

Denoting $\alpha = -\langle \boldsymbol{\mu}_0, \boldsymbol{\mu} \rangle_{\mathcal{L}}$, we get

$$\begin{aligned}
\mathrm{d}\,\mathrm{PT}_{\boldsymbol{\mu}_0 \to \boldsymbol{\mu}}(\boldsymbol{\xi}) =& \frac{\partial}{\partial \epsilon}\bigg|_{\epsilon=0} \mathrm{PT}_{\boldsymbol{\mu}_0 \to \boldsymbol{\mu}}(\boldsymbol{v} + \epsilon \boldsymbol{\xi}) \\
=& \frac{\partial}{\partial \epsilon}\bigg|_{\epsilon=0} \left[(\boldsymbol{v} + \epsilon \boldsymbol{\xi}) + \frac{\langle \boldsymbol{\mu} - \alpha \boldsymbol{\mu}_0, \boldsymbol{v} + \epsilon \boldsymbol{\xi} \rangle_{\mathcal{L}}}{\alpha + 1} (\boldsymbol{\mu}_0 + \boldsymbol{\mu}) \right] \\
=& \boldsymbol{\xi} + \frac{\langle \boldsymbol{\mu} - \alpha \boldsymbol{\mu}_0, \boldsymbol{\xi} \rangle_{\mathcal{L}}}{\alpha + 1} (\boldsymbol{\mu}_0 + \boldsymbol{\mu}) = \mathrm{PT}_{\boldsymbol{\mu}_0 \to \boldsymbol{\mu}}(\boldsymbol{\xi}).
\end{aligned} \tag{8}$$

and see that each basis element $\boldsymbol{\xi}_k$ is mapped by $\mathrm{d}\,\mathrm{PT}_{\boldsymbol{\mu}_0 \to \boldsymbol{\mu}}$ to

$$(\mathrm{PT}_{\boldsymbol{\mu}_0 \to \boldsymbol{\mu}}(\boldsymbol{\xi}_1), \mathrm{PT}_{\boldsymbol{\mu}_0 \to \boldsymbol{\mu}}(\boldsymbol{\xi}_2) \ \cdots, \ \mathrm{PT}_{\boldsymbol{\mu}_0 \to \boldsymbol{\mu}}(\boldsymbol{\xi}_n))$$

Because parallel transport is a norm preserving map, $\|\mathrm{PT}_{\boldsymbol{\mu}_0 \to \boldsymbol{\mu}}(\boldsymbol{\xi})\|_{\mathcal{L}} = 1$. That is,

$$\det \left(\frac{\partial\, \mathrm{PT}_{\boldsymbol{\mu}_0 \to \boldsymbol{\mu}}(\boldsymbol{v})}{\partial \boldsymbol{v}} \right) = 1. \tag{9}$$

B. Visual Examples of Hyperbolic Wrapped Distribution $\mathcal{G}(\boldsymbol{\mu}, \Sigma)$

Figure 1 shows examples of hyperbolic wrapped distribution $\mathcal{G}(\boldsymbol{\mu}, \Sigma)$ with various μ and Σ. We plotted the log-density of these distributions by heatmaps. We designate the $\boldsymbol{\mu}$ by the $\times$ mark. The right side of these figures expresses their log-density on the Poincaré ball model, and the left side expresses the same one on the corresponding tangent space.

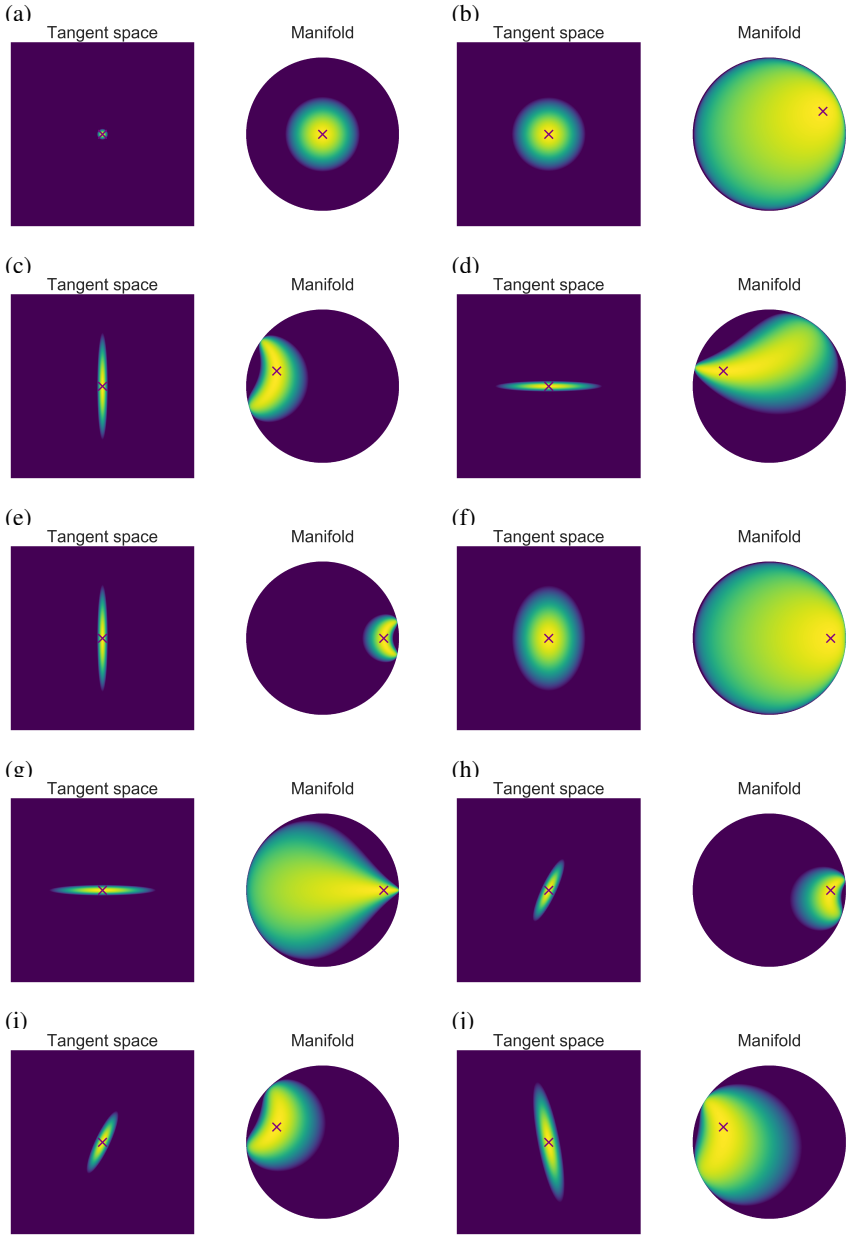


Figure 1: Visual examples of hyperbolic wrapped distribution on $\mathbb{H}^2$. Log-density is illustrated on $\mathcal{B}^2$ by translating each point from $\mathbb{H}^2$ for clarity. We designate the origin of hyperbolic space by the $\times$ mark.

C. Additional Numerical Evaluations

C.1. Synthetic Binary Tree

We qualitatively compared the learned latent space of Vanilla and Hyperbolic VAEs. Figure 2 shows the embedding vectors of the synthetic binary tree dataset on the two-dimensional latent space. We evaluated the latent space of Vanilla VAE with $\beta = 0.1, 1.0, 2.0$, and 3.0, and Hyperbolic VAE. Note that the hierarchical relations in the original tree were **not** used during the training phase. Red points are the embeddings of the noiseless observations. As we mentioned in the main text, we evaluated the correlation coefficient between the Hamming distance on the data space and the hyperbolic (Euclidean for Vanilla VAEs) distance on the latent space. Consistently with this metric, the latent space of the Hyperbolic VAE captured the hierarchical structure inherent in the dataset well. In the comparison between Vanilla VAEs, the latent space captured the hierarchical structure according to increase the β. However, the posterior distribution of the Vanilla VAE with $\beta = 3.0$ collapsed and lost the structure. Also, the blue points are the embeddings of noisy observation, and pink $\times$ represents the origin of the latent space. In latent space of Vanilla VAEs, there was bias in which embeddings of noisy observations were biased to the center side.

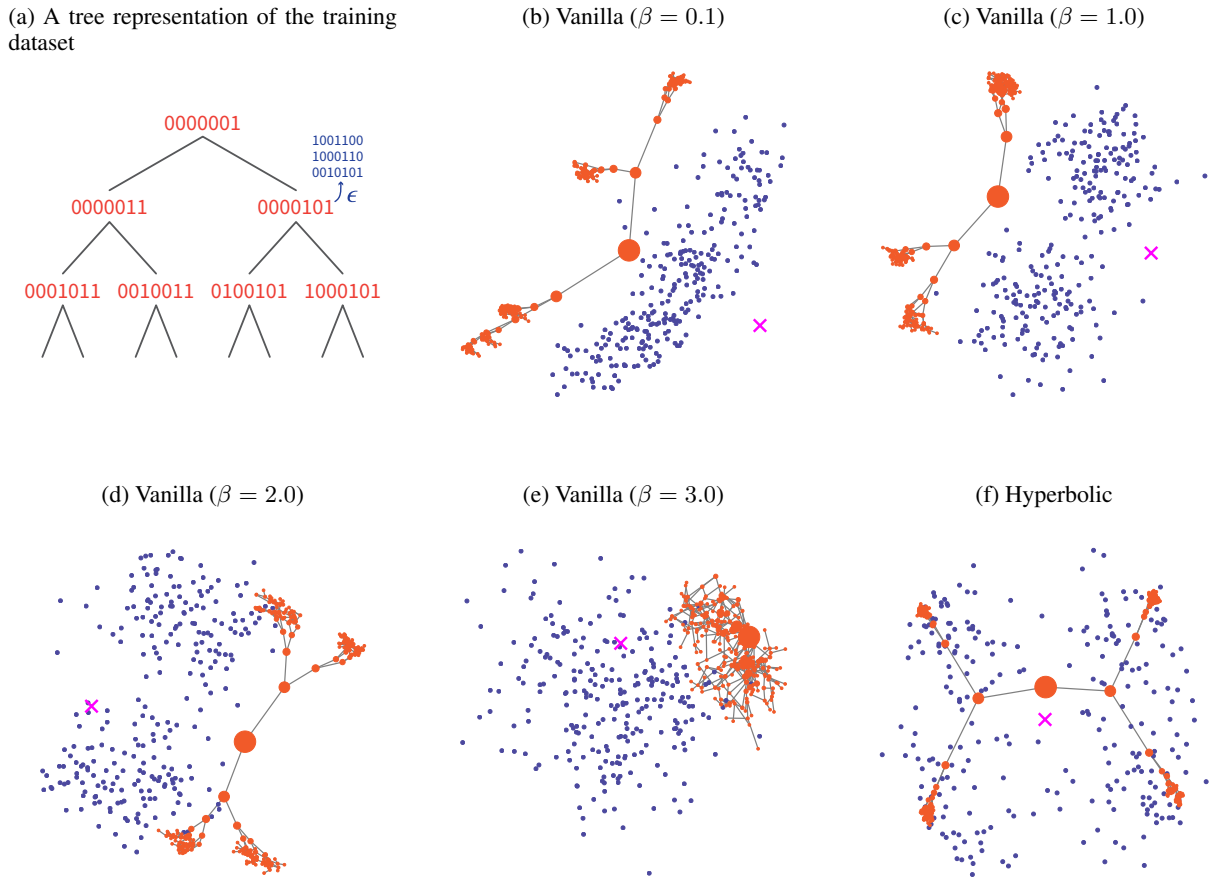


Figure 2: The visual results of Vanilla and Hyperbolic VAEs applied to an artificial dataset generated by applying a random perturbation to a binary tree. The visualization is being done in the Poincaré ball. Red points are the embeddings of the original tree, and the blue points are the embeddings of all other points in the dataset. Pink $\times$ represents the origin of hyperbolic space. Note that the hierarchical relations in the original tree was **not** used during the training phase.

C.2. MNIST

	Vannila VAE		Hyperbolic VAE	
n	ELBO	LL	ELBO	LL
2	$-145.53_{\pm.65}$	$-140.45_{\pm.47}$	$-143.23_{\pm 0.63}$	$\mathbf{-138.61_{\pm 0.45}}$
5	$-111.32_{\pm.38}$	$-105.78_{\pm.51}$	$-111.09_{\pm 0.39}$	$\mathbf{-105.38_{\pm 0.61}}$
10	$-92.49_{\pm.52}$	$\mathbf{-86.25_{\pm.52}}$	$-93.10_{\pm 0.26}$	$-86.40_{\pm 0.28}$
20	$-85.17_{\pm.40}$	$\mathbf{-77.89_{\pm.36}}$	$-88.28_{\pm 0.34}$	$-79.23_{\pm 0.20}$

Table 1: Quantitative comparison of Hyperbolic VAE against Vanilla VAE on the MNIST dataset in terms of ELBO and log-likelihood (LL) for several values of latent space dimension n. LL was computed using 500 samples of latent variables. We calculated the mean and the ± 1 SD with five different experiments.

We showed the numerical performance of Vanilla and Hyperbolic VAEs for MNIST data in the main text in terms of the log-likelihood. In this section, we also show the evidence lower bound for the same dataset in Table 1.

C.3. Atari 2600 Breakout

To evaluate the performance of Hyperbolic VAE for hierarchically organized dataset according to time development, we applied our Hyperbolic VAE to a set of trajectories that were explored by an agent with a trained policy during multiple episodes of Breakout in Atari 2600. We used a pretrained Deep Q-Network to collect trajectories, and Figure 3 shows examples of observed screens.

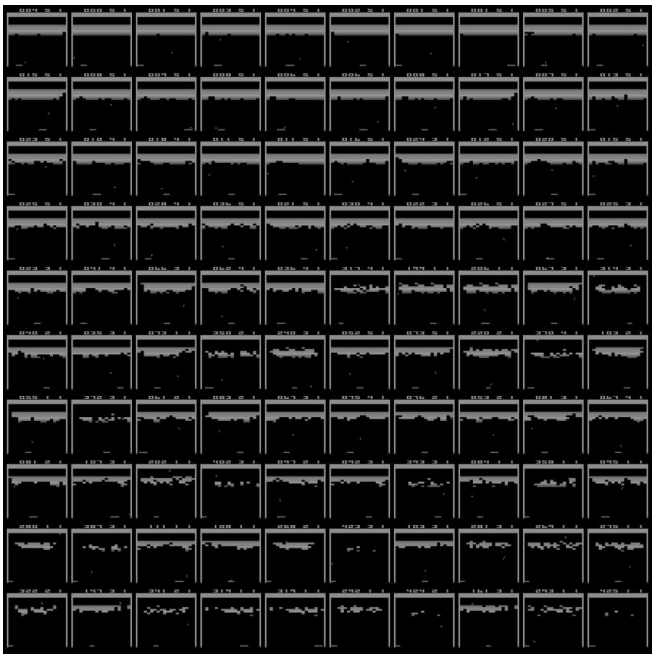
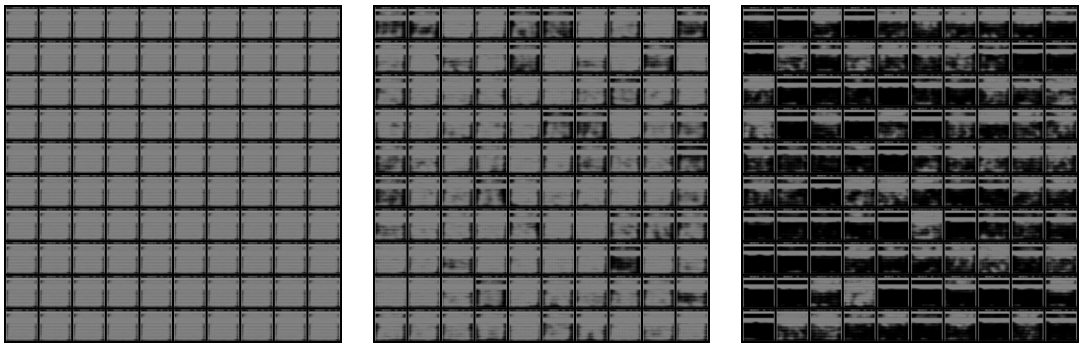


Figure 3: Examples of observed screens in Atari 2600 Breakout.

We showed three trajectories of samples from the prior distribution with the scaled norm for both models in the main text. We also visualize more samples in Figure 4 and 5. For both models, we generated samples with $\|\tilde{\boldsymbol{v}}\|_2 = 0$, 1, 2, 3, 5, and 10.

Vanilla VAE tended to generate oversaturated images when the norm $\|\tilde{\boldsymbol{v}}\|$ was small. Although the model generated several images which include a small number of blocks as the norm increases, it also generated images with a constant amount of blocks even $\|\tilde{\boldsymbol{v}}\| = 10$. On the other hand, the number of blocks contained in the generated image of Hyperbolic VAE gradually decreased according to the norm.

(a) $\|\tilde{\boldsymbol{v}}\|_2 = 0$ (b) $\|\tilde{\boldsymbol{v}}\|_2 = 1$ (c) $\|\tilde{\boldsymbol{v}}\|_2 = 2$



(d) $\|\tilde{\boldsymbol{v}}\|_2 = 3$ (e) $\|\tilde{\boldsymbol{v}}\|_2 = 5$ (f) $\|\tilde{\boldsymbol{v}}\|_2 = 10$

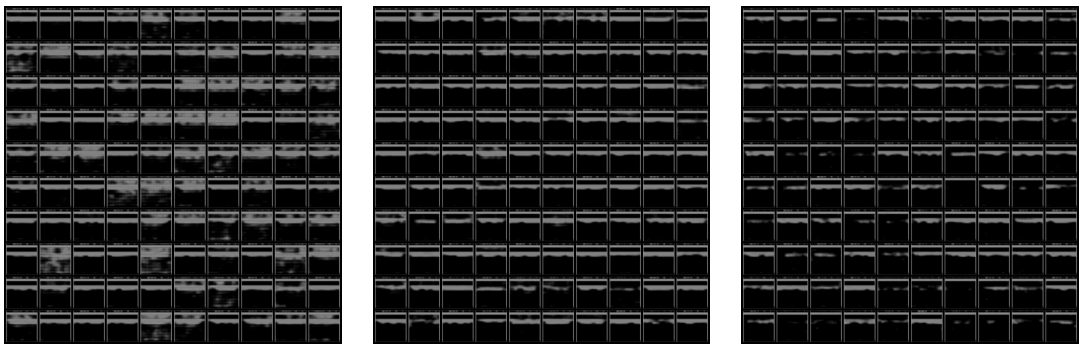
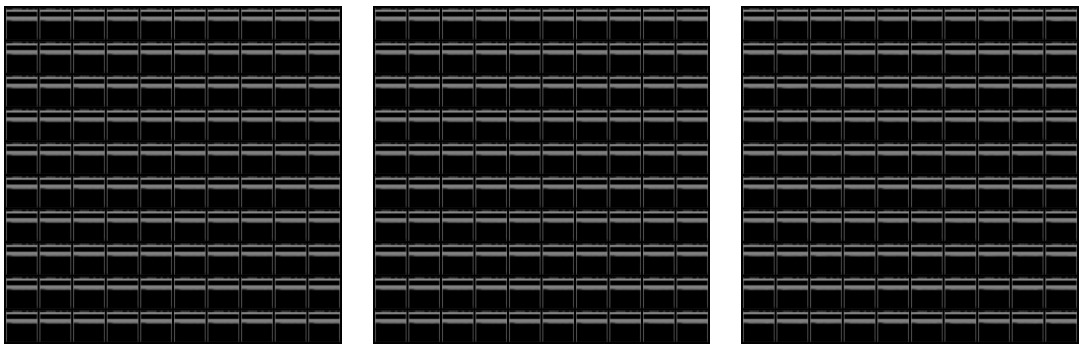


Figure 4: Images generated by Vanilla VAE with constant norm $\|\tilde{\boldsymbol{v}}\|_2 = a$.

(a) $\|\tilde{\boldsymbol{v}}\|_2 = 0$ (b) $\|\tilde{\boldsymbol{v}}\|_2 = 1$ (c) $\|\tilde{\boldsymbol{v}}\|_2 = 2$



(d) $\|\tilde{\boldsymbol{v}}\|_2 = 3$ (e) $\|\tilde{\boldsymbol{v}}\|_2 = 5$ (f) $\|\tilde{\boldsymbol{v}}\|_2 = 10$

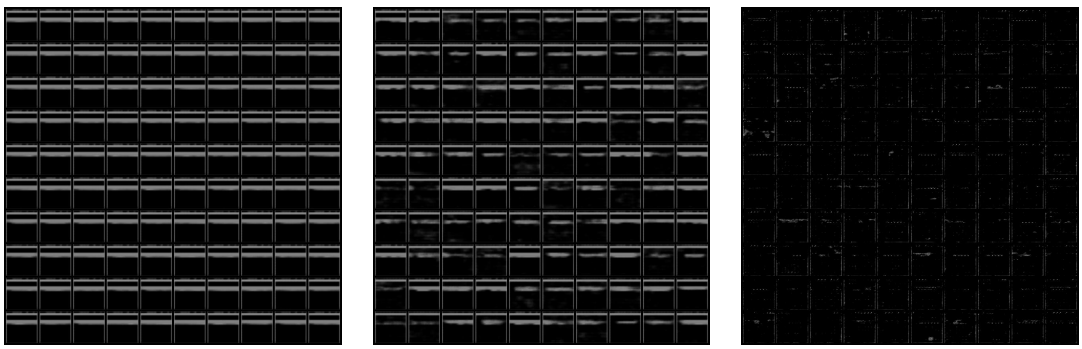


Figure 5: Images generated by Hyperbolic VAE with constant norm $\|\tilde{\boldsymbol{v}}\|_2 = a$.

C.4. Word Embeddings

We showed the experimental results of probabilistic word embedding models with diagonal variance in the main text. Table 2 shows the same comparison with the reference model by Nickel & Kiela (2017). We also show the results with unit variance (Table 3). When the dimensions of the latent variable are small, the performance of the model on hyperbolic space did not deteriorate much by changing the variance from diagonal to unit. However, the same change dramatically worsened the performance of the model on Euclidean space.

	Euclid		Hyperbolic		Nickel & Kiela (2017)	
n	MAP	Rank	MAP	Rank	MAP	Rank
5	$0.296_{\pm.006}$	$25.09_{\pm.80}$	$\mathbf{0.506_{\pm.017}}$	$\mathbf{20.55_{\pm1.34}}$	0.823	4.9
10	$0.778_{\pm.007}$	$\mathbf{4.70_{\pm.05}}$	$\mathbf{0.795_{\pm.007}}$	$5.07_{\pm.12}$	0.851	4.02
20	$0.894_{\pm.002}$	$\mathbf{2.23_{\pm.03}}$	$\mathbf{0.897_{\pm.005}}$	$2.54_{\pm.20}$	0.855	3.84
50	$0.942_{\pm.003}$	$1.51_{\pm.04}$	$\mathbf{0.975_{\pm.001}}$	$\mathbf{1.19_{\pm.01}}$	0.86	3.98
100	$0.953_{\pm.002}$	$1.34_{\pm.02}$	$\mathbf{0.978_{\pm.002}}$	$\mathbf{1.15_{\pm.01}}$	0.857	3.9

Table 2: Experimental results of the reconstruction performance on the transitive closure of the WordNet noun hierarchy for several latent space dimension n. We calculated the mean and the ± 1 SD with three different experiments.

	Euclid		Hyperbolic	
n	MAP	Rank	MAP	Rank
5	$0.217_{\pm.008}$	$55.28_{\pm3.54}$	$\mathbf{0.529_{\pm.010}}$	$\mathbf{22.38_{\pm.70}}$
10	$0.698_{\pm.030}$	$6.54_{\pm.65}$	$\mathbf{0.771_{\pm.006}}$	$\mathbf{5.89_{\pm.29}}$
20	$0.832_{\pm.016}$	$3.08_{\pm.16}$	$\mathbf{0.862_{\pm.002}}$	$\mathbf{2.80_{\pm.13}}$
50	$\mathbf{0.910_{\pm.006}}$	$\mathbf{1.78_{\pm.071}}$	$0.903_{\pm.003}$	$1.94_{\pm.03}$
100	$0.882_{\pm.008}$	$4.75_{\pm2.01}$	$\mathbf{0.884_{\pm.003}}$	$\mathbf{2.57_{\pm.09}}$

Table 3: Experimental results of the word embedding models with unit variance on the WordNet noun dataset. We calculated the mean and the ± 1 SD with three different experiments.

D. Network Architecture

Table 4 shows the network architecture that we used in Breakout experiments. We evaluated Vanilla and Hyperbolic VAEs with a DCGAN-based architecture (Radford et al., 2015) with the kernel size of the convolution and deconvolution layers as 3. We used leaky ReLU nonlinearities for the encoder and ReLU nonlinearities for the decoder. We set the latent space dimension as 20. We gradually increased β from 0.1 to 4.0 linearly during the first 30 epochs. To ensure the initial embedding vector close to the origin, we initialized γ for the batch normalization layer (Ioffe & Szegedy, 2015) of the encoder as 0.1. We modeled the probability distribution of the data space $p(\boldsymbol{x}|\boldsymbol{z})$ as Gaussian, so the decoder output a vector twice as large as the original image.

Encoder	
Layer	Size
Input	$80 \times 80 \times 1$
Convolution	$80 \times 80 \times 16$
BatchNormalization	
Convolution	$40 \times 40 \times 32$
BatchNormalization	
Convolution	$40 \times 40 \times 32$
BatchNormalization	
Convolution	$20 \times 20 \times 64$
BatchNormalization	
Convolution	$20 \times 20 \times 64$
BatchNormalization	
Convolution	$10 \times 10 \times 64$
Linear	$2n$

Decoder	
Layer	Size
Linear	$10 \times 10 \times 64$
BatchNormalization	
Deconvolution	$20 \times 20 \times 32$
BatchNormalization	
Convolution	$20 \times 20 \times 32$
BatchNormalization	
Deconvolution	$40 \times 40 \times 16$
BatchNormalization	
Convolution	$40 \times 40 \times 16$
Deconvolution	$80 \times 80 \times 2$
Convolution	$80 \times 80 \times 2$

Table 4: Network architecture for Atari 2600 Breakout dataset.

References

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning*, volume 37, pp. 448–456, 2015.

Nickel, M. and Kiela, D. Poincaré embeddings for learning hierarchical representations. In *Advances in Neural Information Processing Systems 30*, pp. 6338–6347. 2017.

Radford, A., Metz, L., and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *CoRR*, abs/1511.06434, 2015.