Exploring the Poincaré Ellipsis

Samuel G. Fadel [*1], Tino Paulsen[*1], and Ulf Brefeld[1]

Leuphana University, Lüneburg, Germany
{samuel.fadel,tino.paulsen,ulf.brefeld}@leuphana.de

Abstract. Exploring different geometries has shown to be useful for exploiting inherent properties of data at hand, becoming attractive to compute embeddings therein. For example, hyperbolic geometry shows superior performance when embedding hierarchical data. This suggests that the ability to explore different geometries for embedding data might be an important consideration, just as the models and algorithms used to perform the embedding. However, utilising non-Euclidean geometries to embed data can sometimes be a laborious task, as the whole process needs to be first analytically derived for each geometry chosen (e.g., spherical, hyperbolic) and then adapted to it. This discourages the exploration of richer spaces to embed data in. Using the framework of Riemannian manifolds, we explore a new choice of geometry. We consider a modified version of the Poincaré disc to accept individual (per-dimension) curvatures instead of a single global one, resulting in what we refer to as the *Poincaré ellipsis*. We experiment with link prediction on graph nodes embedded onto these new spaces, showing the performance implications and highlighting the ease to explore new variants of hyperbolic geometry.

Keywords: Riemannian manifolds · non-Euclidean geometries · variational graph autoencoder

1 Introduction

Learning embeddings for data is a useful way of devising models or algorithms that extract useful information from this data into a usable representation for later learning-related downstream tasks. Since most of machine learning is built upon Euclidean geometry, it is usually expected that those embeddings are computed therein. This, however, must not be the case. Existing work has depicted the usefulness of different geometries for embeddings, as shown for hyperbolic geometry [10, 12], or for spherical geometry [3]. The fit between data and geometry generally leads to improved performance, and it has been shown that hyperbolic geometry is especially suited for hierarchical data, Euclidean geometry fits grid-like data and spherical geometry suits cyclical data.

This encourages research to investigate not only canonical forms of those geometries, but also variants. Currently explored geometries usually rely on a

[*] Equal contribution. This project was partially funded by the German Federal Ministry of Education and Research (BMBF, 16KIS1205).

global notion of curvature, which is 0 in the Euclidean case, representing a flat structure, while being negative in the hyperbolic and positive in the spherical case. A discouraging aspect of exploring new geometries is the usual difficulty involved in tweaking the whole learning process around a specific choice of geometry. The framework of Riemannian geometry describes the geometry of manifolds by their metric tensor. This can be utilised to explore new geometries without analytically deriving all involved concepts like, but not limited to, the exponential and logarithmic maps.

As an exemplary application, we propose to modify the Poincaré disc model to accept individual curvatures for each dimension instead of one global curvature. This will result in a variant of the well-known hyperbolic model, which we call the *Poincaré ellipsis*, as different curvature values result in different radii, resembling an elliptical shape. In order to do this, we recap some concepts from Riemannian geometry. Empirically, using a variational graph autoencoder [7], we show how to embed node representations of graph data into this new geometry.

Related work Both [10, 12] introduced a variant of the variational autoencoder using the Poincaré ball as a latent space. We employ a variational graph autoencoder and introduce a latent space with novel geometry.

The approach from [14] is to develop a mixed-curvature latent space. Mixed here means different curvatures for components, a component being a manifold and the latent space being a product of manifolds. In contrast, we propose to use with one manifold instead of a product of manifolds.

[1, 15] showed that training neural networks on purely Euclidean geometry can induce changes in the curvature of the latent space, favouring the use of non-Euclidean tools. Instead of inducing a different geometry using a seemingly Euclidean model, we directly approach training with non-Euclidean geometries.

2 Riemannian geometry

For a more extensive introduction to these concepts we confer to [9], notation-wise we follow [10]. A manifold $\mathcal{M}$ is always locally resembling Euclidean space, Hausdorff and second countable. Furthermore we assume smooth manifolds. For every point $\boldsymbol{z} \in \mathcal{M}$ there is a tangent space $\mathcal{T}_{\boldsymbol{z}}\mathcal{M}$, which is a real vector space of same dimensionality as $\mathcal{M}$ containing all the possible directions of tangential movement. A Riemannian metric $\mathfrak{g}(\boldsymbol{z})$ assigns the tangent space of each point $\boldsymbol{z}$ a smoothly varying inner product:

$$\mathfrak{g}(\boldsymbol{z}) = \langle \cdot, \cdot \rangle_{\boldsymbol{z}} : \mathcal{T}_{\boldsymbol{z}}\mathcal{M} \times \mathcal{T}_{\boldsymbol{z}}\mathcal{M} \to \mathbb{R}.$$

The Riemannian metric can be rewritten to a matrix representation $G(\boldsymbol{z})$ as follows:

$$\forall \boldsymbol{u}, \boldsymbol{v} \in \mathcal{T}_{\boldsymbol{z}}\mathcal{M}, \langle \boldsymbol{u}, \boldsymbol{v} \rangle_{\boldsymbol{z}} = \mathfrak{g}(\boldsymbol{z})(\boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{u}^{\intercal} G(\boldsymbol{z}) \boldsymbol{v}.$$

A manifold $\mathcal{M}$ equipped with a Riemannian metric $\mathfrak{g}_{\boldsymbol{z}}$ is called a Riemannian manifold $(\mathcal{M}, \mathfrak{g}_{\boldsymbol{z}})$, which is sufficient to describe the local geometry. From this, concepts of global geometry can be constructed. The inner product gives rise to a norm on $\mathcal{T}_{\boldsymbol{z}}\mathcal{M}$ given by $\|\cdot\|_{\boldsymbol{z}} = \sqrt{\langle\cdot,\cdot\rangle_{\boldsymbol{z}}}$, the metric tensor allows for a measure $d\mathcal{M}(\boldsymbol{z}) = \sqrt{|G(\boldsymbol{z})|}d\boldsymbol{z}$ with $d\boldsymbol{z}$ being the Lebesgue measure. For constructing the shortest-length paths, which generalise the notion of a straight line in Euclidean space, typically a curve is parametrised. Given a curve $\gamma : t \mapsto \gamma(t) \in \mathcal{M}$ its length is given by

$$L(\gamma) = \int_0^1 \|\gamma'(t)\|_{\gamma(t)}^{1/2}. \tag{1}$$

Now take $\gamma^* = \arg\min L(\gamma)$ with two points $\boldsymbol{z}, \boldsymbol{y} \in \mathcal{M}$ and $\gamma(0) = \boldsymbol{z}, \gamma(1) = \boldsymbol{y}$. The result γ^* is the geodesic from $\boldsymbol{z}$ to $\boldsymbol{y}$. There are different methods to find the geodesic. Analytical direct solutions are known for some models, but in the general case one can solve the differential equation associated with the geodesics, for this we recommend appendix A of [1]. The geodesic length can be used for defining a global distance $d_{\mathcal{M}}(\boldsymbol{z}, \boldsymbol{y}) = \inf L(\gamma)$. To move along a geodesic is called following the exponential map. For every starting point $\boldsymbol{z}$ with initial velocity $\boldsymbol{v}$ there is a unique geodesic γ with $\gamma(0)$ and $\gamma'(0) = \boldsymbol{v}$. If the exponential map is defined for every point $\boldsymbol{z} \in \mathcal{M}$, then $\mathcal{M}$ is geodesically complete. Given two ponts $\boldsymbol{z}, \boldsymbol{y} \in \mathcal{M}$, the logarithmic map moves from $\boldsymbol{y}$ to $\boldsymbol{z}$ and computes the velocity $\boldsymbol{v}$ at point $\boldsymbol{z}$.

3 Embedding data into a Riemannian manifold

We take a variational autoencoder (VAE) [13] as a way of embedding arbitrary data into a lower-dimensional Euclidean space to illustrate an analogous process with any Riemannian manifold described by a smoothly-changing metric tensor.

Without taking its statistical perspective into account, one can consider a simplified view of the encoding process of a VAE as follows. First, it maps a given data point $\boldsymbol{x}_i$ using an encoder neural network into the parameters of a Gaussian distribution. Let $\mathcal{Z}$ denote the set of latent space points, $\boldsymbol{\mu}_i$ and $\boldsymbol{\sigma}_i$ denote the parameters of the Gaussian modeled by the encoder (approximate posterior). In Euclidean space, one can consider this process done and proceed to decoding, which consists of drawing a sample (or multiple) $\boldsymbol{z}_i \sim N(\boldsymbol{\mu}_i, \mathrm{diag}(\boldsymbol{\sigma}_i^2))$, then mapping the point(s) back into input space via the decoder neural network.

Inspired by previous approaches [10, 12], we consider mapping into a Riemannian manifold as *moving around* in that manifold. In other words, the encoder and decoder do not directly place points in the manifold, but rather start at the origin and work with the velocities (i.e., points in the tangent space at the origin) from there, leveraging the exponential map as the mechanism to compute the final location of the point. In addition, one needs to also consider the sampling process described above, which should also behave differently for a distribution defined on the manifold. Statistically, we consider the so-called *wrapped* Gaussian distribution, the result of "wrapping" a Gaussian distribution into a Riemannian manifold by means of a diffeomorphism, also used in earlier work as the Riemannian analogous of a Gaussian distribution. We note this distribution is not to be confused with the Riemannian Gaussian distribution, which is slightly different and more native to Riemannian geometry.

We start by letting $\boldsymbol{\mu}_i = \exp_0(\boldsymbol{v}_i)$, where $\boldsymbol{v}_i$ denotes the mean predicted by the encoder instead and $\exp_0$ is the exponential map at the origin. Then, $\boldsymbol{z}_i$ is computed in two steps. First, let $\boldsymbol{\sigma}_i$ work as a standard deviation, using it to sample a new velocity $\boldsymbol{v}_i^{(\sigma)} \sim N(0, \operatorname{diag}(\boldsymbol{\sigma}_i^2))$ at the tangent space of $\boldsymbol{\mu}_i$. With the same analogy as before, we use that second velocity to find the final sampled point, $\boldsymbol{z}_i$, by letting $\boldsymbol{z}_i = \exp_{\boldsymbol{\mu}_i}(\boldsymbol{v}_i^{(\sigma)})$. This means that the encoded point $\boldsymbol{z}_i$ is found by moving from the predicted mean $\boldsymbol{\mu}_i$ by a certain standard deviation $\boldsymbol{\sigma}_i$, represented concretely as the velocity $\boldsymbol{v}_i^{(\sigma)}$. Usually, moving from the origin to $\boldsymbol{\mu}$ is done using parallel transport because this preserves the norm of the transported vector. In our case, apart from parallel transport being more complex to approximate than the exponential map, preserving the norm is not intended, as we sample from $\boldsymbol{\mu}$.

Using the exponential maps to move around, we ensure that every point produced in the process properly exists in the manifold and follows its geometry, while backpropagating through the exponential map is enough for the neural networks used in the process to learn to leverage this to embed points into the manifold. For decoding, we use the analogous reverse operator of the exponential map, letting every $\boldsymbol{v}_i = \log_0(\boldsymbol{z}_i)$ and then proceeding with those velocities as input to the decoder.

The advantage of approaching the embedding of data into Riemannian manifolds as proposed here is to be able to handle any Riemannian manifold by simply defining the operations in terms of geodesics, exponential, and logarithmic maps. This also implies that the exponential map and logarithmic map must not be analytically known, which is usually the main technical reason when considering new Riemannian manifolds to use as the target embedding space. For this reason, we solve the differential equations associated with the geodesics with `stochman`, a package to numerically approximate geodesics, exponential maps and logarithmic maps [4].

4 From Poincaré disc to Poincaré ellipsis

There are different models to construct the d-dimensional hyperbolic space $\mathbb{H}^d$ such as the hyperboloid model, the Poincaré half-plane model, the Beltrami-Klein model, the southern hemisphere model and the Poincaré disc model [2]. For $d > 2$ instead of Poincaré disc the model is also called Poincaré ball model.

Poincaré ball The Poincaré ball model is a d-dimensional Riemannian manifold with constant negative curvature c defined as $\mathbb{B}_c^d = (\mathcal{B}_c^d, \mathfrak{g}_b^c)$ with $\mathcal{B}_c^d$ being the open ball of radius $1/\sqrt{c}$ and $\mathfrak{g}_b^c$, the Poincaré ball metric tensor, given by

$$\mathfrak{g}_b^c(\boldsymbol{z}) = (\lambda_{\boldsymbol{z}}^c)^2 \ \mathfrak{g}_e(\boldsymbol{z}), \quad \lambda_{\boldsymbol{z}}^c = \frac{2}{1 - c\|\boldsymbol{z}\|^2}, \tag{2}$$

where $\lambda_{\boldsymbol{z}}^{c}$ is the conformal factor and $\mathfrak{g}_e$ is the Euclidean metric tensor. The Euclidean metric tensor is the usual dot product or in a matrix representation the identity matrix with diagonal $\mathbb{1}$. Following is a depiction of the matrix representation of $\mathfrak{g}_b^c(\boldsymbol{z})$:

$$G_b^c(\boldsymbol{z}) = \begin{pmatrix} (\lambda_{\boldsymbol{z}}^{c})^2 & \cdots & 0 \\ & \ddots & \\ \vdots & (\lambda_{\boldsymbol{z}}^{c})^2 & \vdots \\ & \ddots & \\ 0 & \cdots & (\lambda_{\boldsymbol{z}}^{c})^2 \end{pmatrix}$$

The distance function, exponential map and logarithmic map for the Poincaré ball are analytically known, cf. [10].

Poincaré ellipsis In the Poincaré ball the curvature value c of the metric tensor is a scalar. We propose a novel Riemannian manifold that works with a vector $\mathbf{c} = (c_1, \ldots, c_d)$ instead of a scalar, which we will refer to by $\mathbb{P}_{\mathbf{c}}^d = (\mathcal{E}_{\mathbf{c}}^d, \mathfrak{g}_p^{\mathbf{c}})$, where $\mathcal{E}_{\mathbf{c}}^d$ is the ellipsoid with radii $1/\sqrt{c_i}$ depending on the curvature c_i associated with the dimension. Applying the same change from scalar c to vector $\mathbf{c}$ to the metric tensor results in:

$$\mathfrak{g}_p^{\mathbf{c}}(\boldsymbol{z}) = (\lambda_{\boldsymbol{z}}^{\mathbf{c}})^2 \; \mathfrak{g}_e(\boldsymbol{z}), \quad \lambda_{\boldsymbol{z}}^{\mathbf{c}} = \frac{2}{1 - \mathbf{c}\|\boldsymbol{z}\|^2}. \tag{3}$$

The closeness to the metric tensor shown in equation (2) is obvious. It can also be observed in the matrix representation of the $\mathfrak{g}_p^{\mathbf{c}}(\boldsymbol{z})$:

$$G_p^{\mathbf{c}}(\boldsymbol{z}) = \begin{pmatrix} (\lambda_{\boldsymbol{z}}^{c_1})^2 & \cdots & 0 \\ & \ddots & \\ \vdots & (\lambda_{\boldsymbol{z}}^{c_i})^2 & \vdots \\ & \ddots & \\ 0 & \cdots & (\lambda_{\boldsymbol{z}}^{c_d})^2 \end{pmatrix}$$

Even though the change from c to $\mathbf{c}$ is simple, the implications of it are not. One can follow the general technique to parametrise a curve γ and solve equation (1) to use the length of the geodesic as distance. To the best of our knowledge, the analytical solution is not known.

Another implication of $\boldsymbol{c}$ is that the Poincaré ball, which typically has a global radius of $1/\sqrt{c}$, now has this radius split up for each dimension, which is indicated by the metric tensor even without defining the space explicitly. This changes, for $d = 2$, the disc to become an ellipsis or for $d > 2$, the ball to become an ellipsoid. For the ellipsis this can be seen in the plots of the latent spaces in section 5. The different structure of this geometry results in distance becoming larger at different speeds in different dimensions, which suits different rates of volume growth in the latent space.

5 Experiments

Experimentally, we explore the behaviour of the new geometry in a link prediction task on two widely known graph data sets. We consider three geometries as latent space $\mathcal{Z}$: $\mathbb{R}^d$ as Euclidean space, $\mathbb{B}_c^d$ as standard Poincaré ball and $\mathbb{P}_{\mathbf{c}}^d$ as Poincaré ellipsis, thus $\mathcal{Z} \in \{\mathbb{R}^d, \mathbb{B}_c^d, \mathbb{P}_{\mathbf{c}}^d\}$.

As model we choose a VGAE, following the experimental setup by the authors [7]. Let $\tilde{A} \in \mathbb{R}^{n\times n}$ be the normalised adjacency matrix of the input graph with n nodes [8]. As in [7], we use as encoder $X^{(\mu)} = X'\tilde{A}W^{(\mu)}$ and $X^{(\sigma)} = X'\tilde{A}W^{(\sigma)}$, with $X' = \tilde{A}W_0$. The parameter W_0 is $n \times 32$, while both $W^{(\mu)}$ and $W^{(\sigma)}$ are $32 \times d$, resulting in d-dimensional latent spaces. As decoder we also follow the original methodology and use the Euclidean inner product $\langle \boldsymbol{v}_i, \boldsymbol{v}_j\rangle^{\mathbb{E}}$ to predict the probability that the two nodes represented by them are connected after transforming the inner product through a sigmoid function.

Data set We use the CORA [11] and CITESEER [5] data sets. CORA contains information about 2708 publications from 7 classes and 5429 links (citations) between them. CITESEER consists of 3312 publications from 6 classes and 4732 links between them. Again, we follow the data processing procedure from [7].

Experimental setup We utilise $d = 2$ and train the models for 800 epochs, validating every 10 epochs, on a training version of the data with certain links masked, again following [7], optimising the parameters with Adam in its default setting [6]. Our evaluation metrics follow existing similar work [10, 3]: area under the ROC curve (AUC) and average precision (AP). In each scenario, we choose the best model checkpoint in the validation set to be used for testing.

Table 1. AUC and AP results on a link prediction task on CORA and CITESEER. Confidence intervals are computed over X runs.

	CORA		CITESEER	
Models	**AUC**	**AP**	**AUC**	**AP**
$\mathbb{R}^2$–VGAE	$0.67_{\pm 0.02}$	$0.70_{\pm 0.02}$	$0.68_{\pm 0.01}$	$0.73_{\pm 0.01}$
$\mathbb{B}^2_{1.0}$–VGAE	$0.70_{\pm 0.02}$	$0.73_{\pm 0.02}$	$0.69_{\pm 0.02}$	$0.73_{\pm 0.02}$
$\mathbb{P}^2_{(0.8,1.15)}$–VGAE	$0.71_{\pm 0.02}$	$0.73_{\pm 0.01}$	$0.68_{\pm 0.02}$	$0.72_{\pm 0.02}$

Results Preliminary empirical results on link prediction are shown in Table 1. Generally it should be noted that these models are not tuned in an extensive manner. On the CORA data set the hyperbolic models outperform the Euclidean, but show similar performances between the models with global constant and individual curvatures. It could well be possible that the effects of individual curvature are more visible with better tuned models.

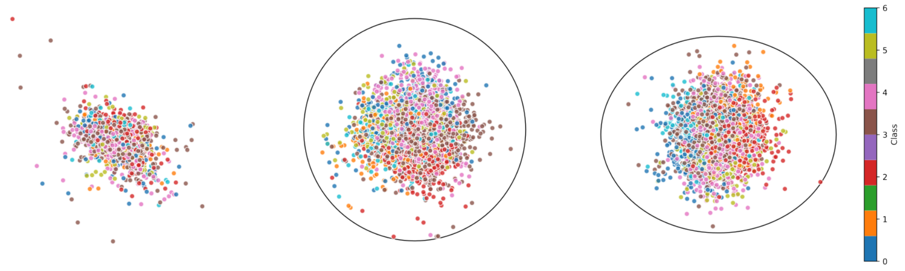

Fig. 1. From left to right: The latent space of a Euclidean space, a Poincaré disc, a Poincaré ellipsis. Data set is CORA.

The CITESEER shows no significant difference in performance between all the models, which hints at the models not revealing their full potential. More tuning and experiments are in need here.

As both CORA and CITESEER are citation networks, they have relatively clear structures in them. More difficult data sets which contain more complicated hierarchies could benefit from the individual curvature model.

The aforementioned change in latent spaces from disc to ellipsis in the case of individual curvatures can be observed in Figure 1. Distributing the curvature over dimensions allows the model to allocate the growth of distance per dimension, which can help with embedding quality.

6 Conclusions and future work

We showed an approach to explore a novel geometry using the framework of Riemannian manifolds, introducing a variant of hyperbolic geometry we refer to as *Poincaré ellipsis*. Individual curvature per dimension allow for different growths in distance in different directions, which can be explored by the models performing embeddings to encode more complex information. The model using the Poincaré ellipsis shows potential to perform on par with the model using the Poincaré ball, hinting at the potentional of improving embedding quality. This seems to adhere to previous insights that the geometry should fit the structure of the data. In many hierarchical data sets the hierarchies are not perfectly regular or behave differently across different children, these data sets should benefit from individual curvatures per dimension.

As motivated earlier, the exploration of new geometries, e.g. modifications to spherical geometries are natural continuations of this work. Additionally, combining our approach with [14], the mixed-curvature VAEs, could allow for even more complex structures in latent spaces. If the components, i.e. the submanifolds, getting multiplied already have non-constant curvature, the resulting manifolds should be even more flexible.

Acknowledgements We thank Jennifer Matthiesen, Kai Neubauer, Yannick Rudolph and Moritz Wohlstein for insightful discussions.

References

1. Arvanitidis, G., Hansen, L.K., Hauberg, S.: Latent space oddity: on the curvature of deep generative models. arXiv preprint arXiv:1710.11379 (2017)
2. Cannon, J.W., Floyd, W.J., Kenyon, R., Parry, W.R., et al.: Hyperbolic geometry. Flavors of geometry **31**(59-115), 2 (1997)
3. Davidson, T.R., Falorsi, L., De Cao, N., Kipf, T., Tomczak, J.M.: Hyperspherical variational auto-encoders. arXiv preprint arXiv:1804.00891 (2018)
4. Detlefsen, N.S., Pouplin, A., Feldager, C.W., Geng, C., Kalatzis, D., Hauschultz, H., González-Duque, M., Warburg, F., Miani, M., Hauberg, S.: Stochman. GitHub. Note: https://github.com/MachineLearningLifeScience/stochman/ (2021)
5. Giles, C.L., Bollacker, K.D., Lawrence, S.: Citeseer: An automatic citation indexing system. In: Proceedings of the third ACM conference on Digital libraries. pp. 89–98 (1998)
6. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
7. Kipf, T.N., Welling, M.: Variational graph auto-encoders. arXiv preprint arXiv:1611.07308 (2016)
8. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: International Conference on Learning Representations (2017)
9. Lee, J.M.: Introduction to Riemannian Manifolds. Springer, 2 edn. (2018)
10. Mathieu, E., Le Lan, C., Maddison, C.J., Tomioka, R., Teh, Y.W.: Continuous hierarchical representations with poincaré variational auto-encoders. Advances in neural information processing systems **32** (2019)
11. McCallum, A.K., Nigam, K., Rennie, J., Seymore, K.: Automating the construction of internet portals with machine learning. Information Retrieval **3**, 127–163 (2000)
12. Nagano, Y., Yamaguchi, S., Fujita, Y., Koyama, M.: A wrapped normal distribution on hyperbolic space for gradient-based learning. In: Chaudhuri, K., Salakhutdinov, R. (eds.) Proceedings of the 36th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 97, pp. 4693–4702. PMLR (09–15 Jun 2019)
13. P Kingma, D., Welling, M., et al.: Auto-encoding variational bayes. In: Proceedings of the International Conference on Learning Representations (ICLR). vol. 1 (2014)
14. Skopek, O., Ganea, O.E., Bécigneul, G.: Mixed-curvature variational autoencoders. In: International Conference on Learning Representations (2020)
15. Zavatone-Veth, J.A., Rubinfien, J.A., Pehlevan, C.: Training shapes the curvature of shallow neural network representations. In: NeurIPS 2022 Workshop on Symmetry and Geometry in Neural Representations (2022)