ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/324182043

Hyperspherical Variational Auto-Encoders

Article · April 2018

CITATIONS
145

READS
25,463

5 authors, including:

Nicola De Cao
University of Amsterdam
27 PUBLICATIONS **1,010** CITATIONS
SEE PROFILE

Thomas Kipf
University of Amsterdam
46 PUBLICATIONS **28,354** CITATIONS
SEE PROFILE

Jakub Mikolaj Tomczak
Vrije Universiteit Amsterdam
122 PUBLICATIONS **3,539** CITATIONS
SEE PROFILE

All content following this page was uploaded by Nicola De Cao on 21 May 2018.

The user has requested enhancement of the downloaded file.

arXiv:1804.00891v1 [stat.ML] 3 Apr 2018

Hyperspherical Variational Auto-Encoders

Tim R. Davidson* **Luca Falorsi*** **Nicola De Cao*** **Thomas Kipf** **Jakub M. Tomczak**

University of Amsterdam

Abstract

The Variational Auto-Encoder (VAE) is one of the most used unsupervised machine learning models. But although the default choice of a Gaussian distribution for both the prior and posterior represents a mathematically convenient distribution often leading to competitive results, we show that this parameterization fails to model data with a latent hyperspherical structure. To address this issue we propose using a von Mises-Fisher (vMF) distribution instead, leading to a hyperspherical latent space. Through a series of experiments we show how such a hyperspherical VAE, or $\mathcal{S}$-VAE, is more suitable for capturing data with a hyperspherical latent structure, while outperforming a normal, $\mathcal{N}$-VAE, in low dimensions on other data types.

1 INTRODUCTION

First introduced by Kingma and Welling (2013); Rezende et al. (2014), the Variational Auto-Encoder (VAE) is an unsupervised generative model that presents a principled fashion for performing variational inference using an auto-encoding architecture. Applying the non-centered parameterization of the variational posterior (Kingma and Welling, 2014), further simplifies sampling and allows to reduce bias in calculating gradients for training. Although the default choice of a Gaussian prior is mathematically convenient, we can show through a simple example that in some cases it breaks the assumption of an *uninformative* prior leading to unstable results. Imagine a dataset on the circle $\mathcal{Z} \subset \mathcal{S}^1$, that is subsequently embedded in $\mathbb{R}^N$ using a transformation f to obtain $f : \mathcal{Z} \to \mathcal{X} \subset \mathbb{R}^N$. Given two hidden units, an autoencoder quickly discovers the latent circle, while a normal VAE becomes highly unstable. This is to be expected as a Gaussian prior is concentrated around the origin, while the KL-divergence tries to reconcile the differences between $\mathcal{S}^1$ and $\mathbb{R}^2$.

The fact that some data types like *directional data* are better explained through spherical representations is long known and well-documented (Mardia, 1975; Fisher et al., 1987), with examples spanning from protein structure, to observed wind directions. Moreover, for many modern problems such as text analysis or image classification, data is often first normalized in a preprocessing step to focus on the directional distribution. Yet, few machine learning methods explicitly account for the intrinsically spherical nature of some data in the modeling process. In this paper, we propose to use the *von Mises-Fisher* (vMF) distribution as an alternative to the Gaussian distribution. This replacement leads to a hyperspherical latent space as opposed to a hyperplanar one, where the Uniform distribution on the hypersphere is conveniently recovered as a special case of the vMF. Hence this approach allows for a truly uninformative prior, and has a clear advantage in the case of data with a hyperspherical interpretation. This was previously attempted by Hasnat et al. (2017), but crucially they do not learn the concentration parameter around the mean, κ.

In order to enable training of the concentration parameter, we extend the *reparameterization trick* for rejection sampling as recently outlined in Naesseth et al. (2017) to allow for n additional transformations. We then combine this with the rejection sampling procedure proposed by Ulrich (1984) to efficiently reparameterize the VAE [1].

We demonstrate the utility of replacing the normal distribution with the von Mises-Fisher distribution for generating latent representations by conducting a range of experiments in three distinct settings. First, we show that

*Equal contribution. . Correspondence to: Nicola De Cao <nicola.decao@student.uva.nl>.

[1] Code freely available on: `https://github.com/nicola-decao/s-vae`

our $\mathcal{S}$-VAEs outperform VAEs with the Gaussian variational posterior ($\mathcal{N}$-VAEs) in recovering a hyperspherical latent structure. Second, we conduct a thorough comparison with $\mathcal{N}$-VAEs on the MNIST dataset through an unsupervised learning task and a semi-supervised learning scenario. Finally, we show that $\mathcal{S}$-VAEs can significantly improve link prediction performance on citation network datasets in combination with a *Variational Graph Auto-Encoder* (VGAE) (Kipf and Welling, 2016).

2 VARIATIONAL AUTO-ENCODERS

2.1 FORMULATION

In the VAE setting, we have a latent variable model for data, where $\mathbf{z} \in \mathbb{R}^M$ denotes latent variables, $\mathbf{x}$ is a vector of D observed variables, and $p_\phi(\mathbf{x}, \mathbf{z})$ is a parameterized model of the joint distribution. Our objective is to optimize the log-likelihood of the data, $\log \int p_\phi(\mathbf{x}, \mathbf{z}) d\mathbf{z}$. When $p_\phi(\mathbf{x}, \mathbf{z})$ is parameterized by a neural network, marginalizing over the latent variables is generally intractable. One way of solving this issue is to maximize the Evidence Lower Bound (ELBO)

$$\log \int p_\phi(\mathbf{x}, \mathbf{z}) d\mathbf{z} \geq \mathbb{E}_{q(\mathbf{z})}[\log p_\phi(\mathbf{x}|\mathbf{z})] + - KL(q(\mathbf{z})||p(\mathbf{z})), \quad (1)$$

where $q(\mathbf{z})$ is the approximate posterior distribution, belonging to a family $\mathcal{Q}$. The bound is tight if $q(\mathbf{z}) = p(\mathbf{z}|\mathbf{x})$, meaning $q(\mathbf{z})$ is optimized to approximate the true posterior. While in theory $q(\mathbf{z})$ should be optimized for every data point $\mathbf{x}$, to make inference more scalable to larger datasets the VAE setting introduces an inference network $q_\psi(\mathbf{z}|\mathbf{x}; \theta)$ parameterized by a neural network that outputs a probability distribution for each data point $\mathbf{x}$. The final objective is therefore to maximize

$$\mathcal{L}(\phi, \psi) = \mathbb{E}_{q_\psi(\mathbf{z}|\mathbf{x};\theta)}[\log p_\phi(\mathbf{x}|\mathbf{z})] + - KL(q_\psi(\mathbf{z}|\mathbf{x}; \theta)||p(\mathbf{z})), \quad (2)$$

In the original VAE both the prior and the posterior are defined as normal distributions. We can further efficiently approximate the ELBO by Monte Carlo estimates, using the *reparameterization trick* (Kingma and Welling, 2013; Rezende et al., 2014). This is done by expressing a sample of $\mathbf{z} \sim q_\psi(\mathbf{z}|\mathbf{x}; \theta)$, as $\mathbf{z} = h(\theta, \varepsilon, \mathbf{x})$, where h is a reparameterization transformation and $\varepsilon \sim s(\varepsilon)$ is some noise random variable independent from θ.

2.2 THE LIMITATIONS OF A GAUSSIAN DISTRIBUTION PRIOR

Low dimensions: origin gravity In low dimensions, the Gaussian density presents a concentrated probability mass around the origin, encouraging points to cluster in the center. This is particularly problematic when the data is divided into multiple clusters. Although an ideal latent space should separate clusters for each class, the normal prior will encourage all the cluster centers towards the origin. An ideal prior would only stimulate the variance of the posterior without forcing its mean to be close to the center. A prior satisfying these properties is a uniform over the entire space. Such a uniform prior, however, is not well defined on the hyperplane.

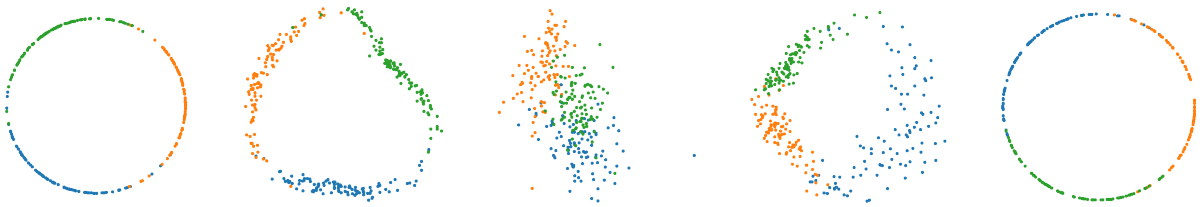
High dimensions: soap bubble effect It is a well-known phenomenon that the standard Gaussian distribution in high dimensions tends to resemble a uniform distribution on the surface of a hypersphere, with the vast majority of its mass concentrated on the hyperspherical shell. Hence it would appear interesting to compare the behavior of a Gaussian approximate posterior with an approximate posterior already naturally defined on the hypersphere. This is also motivated from a theoretical point of view, since the Gaussian definition is based on the L_2 norm that suffers from the curse of dimensionality.

2.3 BEYOND THE HYPERPLANE

Once we let go of the hyperplanar assumption, the possibility of a uniform prior on the hypersphere opens up. Mirroring our discussion in the previous subsection, such a prior would exhibit no pull towards the origin allowing clusters of data to evenly spread over the surface with no directional bias. Additionally, in higher dimensions, the cosine similarity is a more meaningful distance measure than the Euclidean norm.

Manifold mapping In general, exploring VAE models that allow a mapping to distributions in a latent space not homeomorphic to $\mathbb{R}^D$ is of fundamental interest. Consider data lying in a small M-dimensional manifold $\mathcal{M}$, embedded in a much higher dimensional space $\mathcal{X} = \mathbb{R}^N$. For most real data, this manifold will likely not be homeomorphic to $\mathbb{R}^M$. An encoder can be considered as a smooth map $enc : \mathcal{X} \to \mathcal{Z} = \mathbb{R}^D$ from the original space to $\mathcal{Z}$. The restriction of the encoder to $\mathcal{M}$, $enc|_{\mathcal{M}} : \mathcal{M} \to \mathcal{Z}$ will also be a smooth mapping. However since $\mathcal{M}$ is not homeomorphic to $\mathcal{Z}$ if $D \leq M$, then $enc|_{\mathcal{M}}$ cannot be a homeomorphism. That is, there exists no invertible and globally continuous mapping between the coordinates of $\mathcal{M}$ and the ones of $\mathcal{Z}$. Conversely if $D > M$ then $\mathcal{M}$ can be smoothly embedded in $\mathcal{Z}$ for D sufficiently big [2] , such that $enc|_{\mathcal{M}} : \mathcal{M} \to enc|_{\mathcal{M}}(\mathcal{M}) =: emb(\mathcal{M}) \subset \mathcal{Z}$ is a homeomorphism and $emb(\mathcal{M})$ denotes the embedding of

[2] By the Whitney embedding theorem any smooth real M-dimensional manifold can be smoothly embedded in $\mathbb{R}^{2M}$



(a) Original (b) Autoencoder (c) $\mathcal{N}$-VAE (d) $\mathcal{N}$-VAE, $\beta = 0.1$ (e) $\mathcal{S}$-VAE

Figure 1: Plots of the original latent space (a) and learned latent space representations in different settings, where β is a re-scaling factor for weighting the KL divergence. (Best viewed in color)

$\mathcal{M}$. Yet, since $D > M$, when taking random points in the latent space they will most likely *not* be in $emb(\mathcal{M})$ resulting in a poorly reconstructed sample.

The VAE tries to solve this problem by forcing $\mathcal{M}$ to be mapped into an approximate posterior distribution that has support in the entire $\mathcal{Z}$. Clearly, this approach is bound to fail since the two spaces have a fundamentally different structure. This can likely produce two behaviors: first, the VAE could just smooth the original embedding $emb(\mathcal{M})$ leaving most of the latent space empty, leading to bad samples. Second, if we increase the KL term the encoder will be pushed to occupy all the latent space, but this will create instability and discontinuity, affecting the convergence of the model. To validate our intuition we performed a small proof of concept experiment using $\mathcal{M} = \mathcal{S}^1$, which is visualized in Figure 1. Note that as expected the auto-encoder in Figure 1(b) mostly recovers the original latent space of Figure1(a) as there are no distributional restrictions. In Figure 1(c) we clearly observe for the $\mathcal{N}$-VAE that points collapse around the origin due to the KL, which is much less pronounced in Figure 1(d) when its contribution is scaled down. Lastly, the $\mathcal{S}$-VAE almost perfectly recovers the original circular latent space. The observed behavior confirms our intuition.

To solve this problem the best option would be to directly specify a $\mathcal{Z}$ homeomorphic to $\mathcal{M}$ and distributions on $\mathcal{M}$. However, for real data discovering the structure of $\mathcal{M}$ will often be a difficult inference task. Nevertheless, we believe this shows that investigating VAE architectures that map to posterior distributions defined on manifolds different than the Euclidean space is a topic worth to be explored. In that sense, this work represents an initial step in this research direction.

3 REPLACING GAUSSIAN WITH VON MISES-FISHER

3.1 VON MISES-FISHER DISTRIBUTION

The *von Mises-Fisher* (vMF) distribution is often described as the Normal Gaussian distribution on a hypersphere. Analogous to a Gaussian, it is parameterized by $\mu \in \mathbb{R}^m$ indicating the mean direction, and $\kappa \in \mathbb{R}_{\geq 0}$ the concentration around μ. For the special case of $\kappa = 0$, the vMF represents a Uniform distribution. The probability density function of the vMF distribution for a random unit vector $\mathbf{z} \in \mathbb{R}^m$ (or $\mathbf{z} \in \mathcal{S}^{m-1}$) is then defined as

$$q(\mathbf{z}|\mu, \kappa) = \mathcal{C}_m(\kappa) \exp\left(\kappa \mu^T \mathbf{z}\right) \tag{3}$$

$$\mathcal{C}_m(\kappa) = \frac{\kappa^{m/2-1}}{(2\pi)^{m/2} \mathcal{I}_{m/2-1}(\kappa)}, \tag{4}$$

where $||\mu||^2 = 1$, $\mathcal{C}_m(\kappa)$ is the normalizing constant, and $\mathcal{I}_v$ denotes the modified Bessel function of the first kind at order v.

3.2 KL DIVERGENCE

As previously emphasized, one of the main advantages of using the vMF distribution as an approximate posterior is that we are able to place a uniform prior on the latent space. The KL divergence term $KL(\text{vMF}(\mu, \kappa)||U(S^{m-1}))$ to be optimized is:

$$\kappa \frac{\mathcal{I}_{m/2}(k)}{\mathcal{I}_{m/2-1}(k)} + \log \mathcal{C}_m(\kappa) - \log \left(\frac{2(\pi^{m/2})}{\Gamma(m/2)} \right)^{-1}, \tag{5}$$

see Appendix B for complete derivation. Notice that since the KL term does not depend on μ, this is only optimized in the reconstruction term. The above expression cannot be handled by automatic differentiation packages because of the modified Bessel function in $\mathcal{C}_m(\kappa)$. Thus, to optimize this term we derive the gradient with respect to the

Algorithm 1 vMF sampling

Input: dimension m, mean μ, concentration κ
sample $\mathbf{v} \sim U(\mathcal{S}^{m-2})$
sample $\omega \sim g(\omega|\kappa, m) \propto \exp(\omega\kappa)(1-\omega^2)^{\frac{1}{2}(m-3)}$ {acceptance-rejection sampling}
$\mathbf{z}' \leftarrow (\omega; (\sqrt{1-\omega^2})\mathbf{v}^\top)^\top$
$U \leftarrow Householder(\mathbf{e}_1, \mu)$ {Householder transform}
Return: $U\mathbf{z}'$

concentration parameter $\nabla_\kappa KL(\text{vMF}(\mu,\kappa)||U(S^{m-1}))$:

$$\frac{1}{2}k\left(\frac{\mathcal{I}_{m/2+1}(k)}{\mathcal{I}_{m/2-1}(k)}+ - \frac{\mathcal{I}_{m/2}(k)\left(\mathcal{I}_{m/2-2}(k)+\mathcal{I}_{m/2}(k)\right)}{\mathcal{I}_{m/2-1}(k)^2}+1\right), \quad (6)$$

where the modified Bessel functions can be computed without numerical instabilities using the exponentially scaled modified Bessel function.

3.3 SAMPLING PROCEDURE

To sample from the vMF we follow the procedure of Ulrich (1984), outlined in Algorithm 1. We first sample from a vMF $q(\mathbf{z}|\mathbf{e}_1, \kappa)$ with modal vector $\mathbf{e}_1 = (1, 0, \cdots, 0)$. Since the vMF density is uniform in all the $m-2$ dimensional sub-hyperspheres $\{\mathbf{x} \in \mathcal{S}^{m-1}|\mathbf{e}_1^\top\mathbf{x} = \omega\}$, the sampling technique reduces to sampling the value ω from the univariate density $g(\omega|\kappa, m) \propto \exp(\kappa\omega)(1-\omega^2)^{(m-3)/2}, \quad \omega \in [-1, 1]$, using an acceptance-rejection scheme. After getting a sample from $q(\mathbf{z}|\mathbf{e}_1, \kappa)$ an orthogonal transformation $U(\mu)$ is applied such that the transformed sample is distributed according to $q(\mathbf{z}|\mu, \kappa)$. This can be achieved using a Householder reflection such that $U(\mu)\mathbf{e}_1 = \mu$. A more in-depth explanation of the sampling technique can be found in Appendix A.

It is worth noting that the sampling technique does not suffer from the curse of dimensionality, as the acceptance-rejection procedure is only applied to a univariate distribution. Moreover in the case of $\mathcal{S}^2$, the density $g(\omega|\kappa, 3)$ reduces to $g(\omega|\kappa, 3) \propto \exp(k\omega)\mathbb{1}_{[-1,+1]}(\omega)$ which can be directly sampled without rejection.

3.4 N-TRANSFORMATION REPARAMETERIZATION TRICK

While the *reparameterization trick* is easily implementable in the normal case, unfortunately it can only be applied to a handful of distributions. However a recent technique introduced by Naesseth et al. (2017) allows to extend the reparameterization trick to the wide class of distributions that can be simulated using rejection sampling. Dropping the dependence from $\mathbf{x}$ for simplicity, assume the approximate posterior is of the form $g(\omega|\theta)$ and that it can be sampled by making proposals from $r(\omega|\theta)$. If *the proposal distribution can be reparameterized* we can still perform the reparameterization trick. Let $\varepsilon \sim s(\varepsilon)$, and $\omega = h(\varepsilon, \theta)$, a reparameterization of the proposal distribution, $r(\omega|\theta)$. Performing the reparameterization trick for $g(\omega|\theta)$ is made possible by the fundamental lemma proven in (Naesseth et al., 2017):

Lemma 1. *Let f be any measurable function and $\varepsilon \sim \pi(\varepsilon|\theta) = s(\varepsilon)\dfrac{g(h(\varepsilon,\theta)|\theta)}{r(h(\varepsilon,\theta)|\theta)}$ the distribution of the accepted sample. Then:*

$$\mathbb{E}_{\pi(\varepsilon|\theta)}[f(h(\varepsilon,\theta))] = \int f(h(\varepsilon,\theta))\pi(\varepsilon|\theta)d\varepsilon = \int f(\omega)g(\omega|\theta)d\omega = \mathbb{E}_{g(\omega|\theta)}[f(\omega)], \quad (7)$$

Then the gradient can be taken using the log derivative trick:

$$\nabla_\theta\mathbb{E}_{g(\omega|\theta)}[f(\omega)] = \nabla_\theta\mathbb{E}_{\pi(\varepsilon|\theta)}[f(h(\varepsilon,\theta))] = \mathbb{E}_{\pi(\varepsilon|\theta)}[\nabla_\theta f(h(\varepsilon,\theta))] + + \mathbb{E}_{\pi(\varepsilon|\theta)}\left[f(h(\varepsilon,\theta))\nabla_\theta \log\frac{g(h(\varepsilon,\theta)|\theta)}{r(h(\varepsilon,\theta)|\theta)}\right], \quad (8)$$

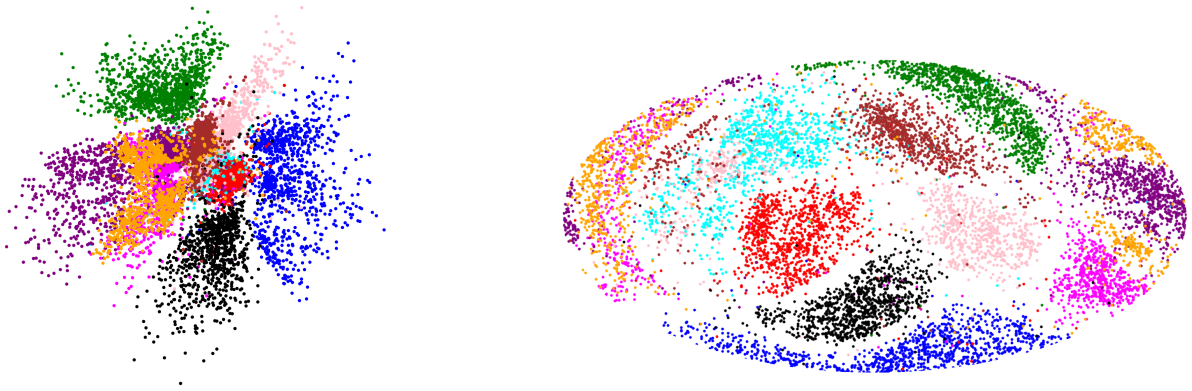
However, in the case of the vMF a different procedure is required. After performing the transformation $h(\varepsilon, \theta)$ and accepting/rejecting the sample, we sample *another* random variable $\mathbf{v} \sim \pi_2(\mathbf{v})$, and then apply a transformation $\mathbf{z} = \mathcal{T}(h(\varepsilon,\theta), \mathbf{v}; \theta)$, such that $\mathbf{z} \sim q_\psi(\mathbf{z}|\theta)$ is distributed as the approximate posterior (in our case a vMF). Effectively this entails applying another reparameterization trick after the acceptance/rejection step. To still be able to perform the reparameterization we show that Lemma 1 fundamentally still holds in this case as well.

Lemma 2. *Let f be any measurable function and $\varepsilon \sim \pi_1(\varepsilon|\theta) = s(\varepsilon)\dfrac{g(h(\varepsilon,\theta)|\theta)}{r(h(\varepsilon,\theta)|\theta)}$ the distribution of the accepted sample. Also let $\mathbf{v} \sim \pi_2(v)$, and $\mathcal{T}$ a transformation that depends on the parameters such that if $\mathbf{z} = \mathcal{T}(\omega, v; \theta)$ with $\omega \sim g(\omega|\theta)$, then $\sim q(\mathbf{z}|\theta)$:*

$$\mathbb{E}_{(\varepsilon,\mathbf{v})\sim\pi_1(\varepsilon|\theta)\pi_2(\mathbf{v})}\left[f\left(\mathcal{T}(h(\varepsilon,\theta),\mathbf{v};\theta)\right)\right] = \int f(\mathbf{z})q(\mathbf{z}|\theta)d\mathbf{z} = \mathbb{E}_{q(\mathbf{z}|\theta)}[f(\mathbf{z})], \quad (9)$$

Proof. See Appendix C. □

With this result we are able to derive a gradient expression similarly as done in equation 8. We refer to Appendix D for a complete derivation.



(a) $\mathbb{R}^2$ latent space of the $\mathcal{N}$-VAE. (b) Hammer projection of $\mathcal{S}^2$ latent space of the $\mathcal{S}$-VAE.

Figure 2: Latent space visualization of the 10 MNIST digits in 2 dimensions of both $\mathcal{N}$-VAE (left) and $\mathcal{S}$-VAE (right). (Best viewed in color)

3.5 BEHAVIOR IN HIGH DIMENSIONS

The surface area of a hypersphere is defined as

$$S(m-1) = r^m \frac{2(\pi^{m/2})}{\Gamma(m/2)} \tag{10}$$

where m is the dimensionality and r the radius. Notice that $S(m-1) \to 0$, as $m \to \infty$. However, even for $m > 20$ we observe a *vanishing surface problem* (see Figure 6 in Appendix E). This could thus lead to unstable behavior of hyperspherical models in high dimensions.

4 RELATED WORK

Extending the VAE The majority of VAE extensions focus on increasing the flexibility of the approximate posterior. This is usually achieved through *normalizing flows* (Rezende and Mohamed, 2015), a class of invertible transformations applied sequentially to an initial reparameterizable density $q_0(\mathbf{z}_0)$, allowing for more complex posteriors. Normalizing flows can be considered orthogonal to our proposed approach. In fact, while allowing for a more flexible posterior, they do not modify the standard normal prior assumption. They could be perfectly combined with $\mathcal{S}$-VAEs allowing for more flexible distributions on the hypersphere.

One approach to obtain a more flexible prior is to use a simple mixture of Gaussians (MoG) prior (Dilokthanakul et al., 2016). The recently introduced VampPrior model (Tomczak and Welling, 2017) outlines several advantages over the MoG and instead tries to learn a more flexible prior by expressing it as a mixture of approximate posteriors. A non-parametric prior is proposed in Nalisnick and Smyth (2017), utilizing a truncated stick-breaking process. Opposite to these approaches, we aim at using a non-informative prior to simplify the inference.

The closest approach to ours is a VAE with a vMF distribution in the latent space used for a sentence generation task by (Guu et al., 2017). While formally this approach is cast as a variational approach, the proposed model does not reparameterize and learn the concentration parameter κ, treating it as a constant value that remains the same for every approximate posterior instead. Critically, as indicated in Equation 5, the KL divergence term only depends on κ therefore leaving κ constant means never explicitly optimizing the KL divergence term in the loss. The method then only optimizes the reconstruction error by adding vMF noise to the encoder output in the latent space to still allow generation. Moreover, using a fixed global κ for *all* the approximate posteriors severely limits the flexibility and the expressiveness of the model.

Non-Euclidean Latent Space In Liu and Zhu (2017), a general model to perform Bayesian inference in Riemannian Manifolds is proposed. Following other Stein-related approaches, the method does not explicitly define a posterior density but approximates it with a number of particles. Despite its generality and flexibility, it requires the choice of a kernel on the manifold and multiple particles to have a good approximation of the posterior distribution. The former is not necessarily straightforward, while the latter quickly becomes computationally unfeasible.

Another approach by Nickel and Kiela (2017), capitalizes on the hierarchical structure present in some data types. By learning the embeddings for a graph in a non-euclidean negative curvature hyperbolical space, they show this topology has clear advantages over embedding these objects in a Euclidean space. Although they did not use a VAE-based approach, that is, they did not build a

Table 1: Summary of results (mean and standard-deviation over 10 runs) of unsupervised model on MNIST. RE and KL correspond respectively to the reconstruction and the KL part of the ELBO. Best results are highlighted only if they passed a student t-test with $p < 0.01$.

Method	$\mathcal{N}$**-VAE**				$\mathcal{S}$**-VAE**			
	LL	$\mathcal{L}[q]$	RE	KL	LL	$\mathcal{L}[q]$	RE	KL
$d = 2$	-135.73$\pm$.83	-137.08$\pm$.83	-129.84$\pm$.91	7.24$\pm$.11	**-132.50**$\pm$.73	-133.72$\pm$.85	-126.43$\pm$.91	7.28$\pm$.14
$d = 5$	-110.21$\pm$.21	-112.98$\pm$.21	-100.16$\pm$.22	12.82$\pm$.11	**-108.43**$\pm$.09	-111.19$\pm$.08	-97.84$\pm$.13	13.35$\pm$.06
$d = 10$	-93.84$\pm$.30	-98.36$\pm$.30	-78.93$\pm$.30	19.44$\pm$.14	**-93.16**$\pm$.31	-97.70$\pm$.32	-77.03$\pm$.39	20.67$\pm$.08
$d = 20$	-88.90$\pm$.26	-94.79$\pm$.19	-71.29$\pm$.45	23.50$\pm$.31	-89.02$\pm$.31	-96.15$\pm$.32	-67.65$\pm$.43	28.50$\pm$.22
$d = 40$	**-88.93**$\pm$.30	-94.91$\pm$.18	-71.14$\pm$.56	23.77$\pm$.49	-90.87$\pm$.34	-101.26$\pm$.33	-67.75$\pm$.70	33.50$\pm$.45

probabilistic generative model of the data interpreting the embeddings as latent variables, this approach shows the merit of explicitly adjusting the choice of latent topology to the data used.

A Hyperspherical Perspective As noted before, a distinction must be made between models dealing with the challenges of intrinsically hyperspherical data like omnidirectional video, and those attempting to exploit some latent hyperspherical manifold. A recent example of the first can be found in Cohen et al. (2018), where *spherical* CNNs are introduced. While flattening a spherical image produces unavoidable distortions, the newly defined convolutions take into account its geometrical properties.

The most general implementation of the second model type was proposed by Gopal and Yang (2014), who introduced a suite of models to improve cluster performance of high-dimensional data based on mixture of vMF distributions. They showed that reducing an object representation to its directional components increases clusterability over standard methods like K-Means or Latent Dirichlet Allocation (Blei et al., 2003).

Specific applications of the vMF can be further found ranging from computer vision, where it is used to infer structure from motion (Guan and Smith, 2017) in spherical video, or structure from texture (Wilson et al., 2014), to natural language processing, where it is utilized in text analysis (Banerjee et al., 2003, 2005) and topic modeling (Banerjee and Basu, 2007; Reisinger et al., 2010).

Additionally, modeling data by restricting it to a hypersphere provides some natural regularizing properties as noted in (Liu et al., 2017). Finally Aytekin et al. (2018) show on a variety of deep auto-encoder models that adding L2 normalization to the latent space during training, i.e. forcing the latent space on a hypersphere, improves clusterability.

5 EXPERIMENTS

In this section, we first perform a series of experiments to investigate the theoretical properties of the proposed $\mathcal{S}$-VAE compared to the $\mathcal{N}$-VAE. In a second experiment, we show how $\mathcal{S}$-VAEs can be used in semi-supervised tasks to create a better separable latent representation to enhance classification. In the last experiment, we show that the $\mathcal{S}$-VAE indeed presents a promising alternative to $\mathcal{N}$-VAEs for data with a non-Euclidean latent representation of low dimensionality, on a link prediction task for three citation networks. All architecture and hyperparameter details are given in Appendix F.

5.1 RECOVERING HYPERSPHERICAL LATENT REPRESENTATIONS

In this first experiment we build on the motivation developed in Subsection 2.3, by confirming with a synthetic data example the difference in behavior of the $\mathcal{N}$-VAE and $\mathcal{S}$-VAE in recovering latent hyperspheres. We first generate samples from a mixture of three vMFs on the circle, $\mathcal{S}^1$, as shown in Figure 1(a), which subsequently are mapped into the higher dimensional $\mathbb{R}^{100}$ by applying a noisy, non-linear transformation. After this, we in turn train an auto-encoder, a $\mathcal{N}$-VAE, and a $\mathcal{S}$-VAE. We further investigate the behavior of the $\mathcal{N}$-VAE, by training a model using a scaled down KL divergence.

Results The resulting latent spaces, displayed in Figure 1, clearly confirm the intuition built in Subsection 2.3. As expected, in Figure 1(b) the auto-encoder is perfectly capable to embed in low dimensions the original underlying data structure. However, most parts of the latent space are not occupied by points, critically affecting the ability to generate meaningful samples.

In the $\mathcal{N}$-VAE setting we observe two types of behaviours, summarized by Figures 1(c) and 1(d). In the first we observe that if the prior is too strong it will force the

Table 2: Summary of results (mean accuracy and standard-deviation over 20 runs) of semi-supervised K-NN on MNIST. Best results are highlighted only if they passed a student t-test with $p < 0.01$.

Method	100		600		1000	
	$\mathcal{N}$-VAE	$\mathcal{S}$-VAE	$\mathcal{N}$-VAE	$\mathcal{S}$-VAE	$\mathcal{N}$-VAE	$\mathcal{S}$-VAE
$d = 2$	$72.6_{\pm 2.1}$	**77.9**$_{\pm 1.6}$	$80.8_{\pm 0.5}$	**84.9**$_{\pm 0.6}$	$81.7_{\pm 0.5}$	**85.6**$_{\pm 0.5}$
$d = 5$	$81.8_{\pm 2.0}$	**87.5**$_{\pm 1.0}$	$90.9_{\pm 0.4}$	**92.8**$_{\pm 0.3}$	$92.0_{\pm 0.2}$	**93.4**$_{\pm 0.2}$
$d = 10$	$75.7_{\pm 1.8}$	**80.6**$_{\pm 1.3}$	$88.4_{\pm 0.5}$	**91.2**$_{\pm 0.4}$	$90.2_{\pm 0.4}$	**92.8**$_{\pm 0.3}$
$d = 20$	$71.3_{\pm 1.9}$	**72.8**$_{\pm 1.6}$	$88.3_{\pm 0.5}$	**89.1**$_{\pm 0.6}$	$90.1_{\pm 0.4}$	**91.1**$_{\pm 0.3}$
$d = 40$	**72.3**$_{\pm 1.6}$	$67.7_{\pm 2.3}$	$88.0_{\pm 0.5}$	$87.4_{\pm 0.7}$	$90.3_{\pm 0.5}$	$90.4_{\pm 0.4}$

posterior to match the prior shape, concentrating the samples in the center. However, this prevents the $\mathcal{N}$-VAE to correctly represent the true shape of the data and creates instability problems for the decoder around the origin. On the contrary, if we scale down the KL term, we observe that the samples from the approximate posterior maintain a shape that reflects the $\mathcal{S}^1$ structure smoothed with Gaussian noise. However, as the approximate posterior differs strongly from the prior, obtaining meaningful samples from the latent space again becomes problematic.

The $\mathcal{S}$-VAE on the other hand, almost perfectly recovers the original dataset structure, while the samples from the approximate posterior closely match the prior distribution. This simple experiment confirms the intuition that having a prior that matches the true latent structure of the data, is crucial in constructing a correct latent representation that preserves the ability to generate meaningful samples.

5.2 EVALUATION OF EXPRESSIVENESS

To compare the behavior of the $\mathcal{N}$-VAE and $\mathcal{S}$-VAE on a data set that does not have a clear hyperspherical latent structure, we evaluate both models on a reconstruction task using dynamically binarized MNIST (Salakhutdinov and Murray, 2008). We analyze the ELBO, KL, negative reconstruction error, and marginal log-likelihood (LL) for both models on the test set. The LL is estimated using importance sampling with 500 sample points (Burda et al., 2015).

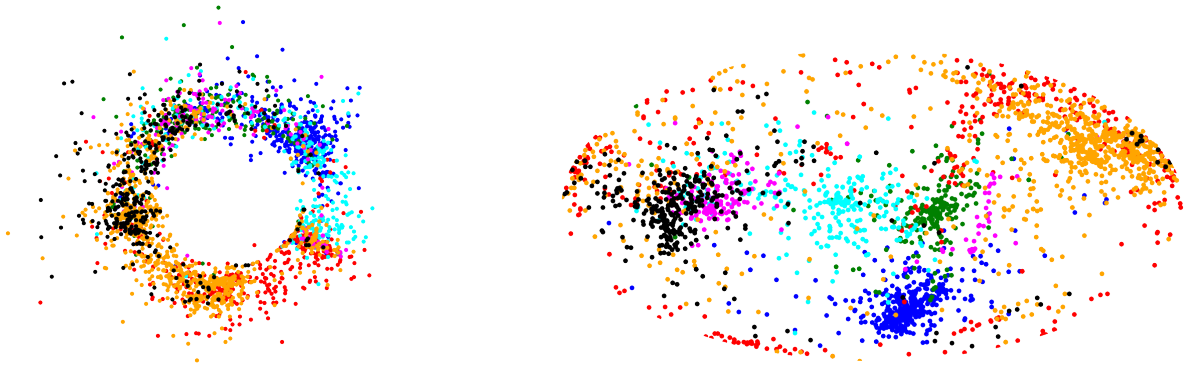
Results Results are shown in Table 1. We first note that in terms of negative reconstruction error the $\mathcal{S}$-VAE outperforms the $\mathcal{N}$-VAE in all dimensions. Since the $\mathcal{S}$-VAE uses a uniform prior, the KL divergence increases more strongly with dimensionality, which results in a higher ELBO. However in terms of log-likelihood (LL) the $\mathcal{S}$-VAE clearly has an edge in low dimensions ($d = 2, 5, 10$) and performs comparable to the $\mathcal{N}$-VAE in $d = 20$. This empirically confirms the hypothesis of Subsection 2.2, showing the positive effect of having a uniform prior in low dimensions. In the absence of any origin pull, the data is able to cluster naturally, utilizing the entire latent space which can be observed in Figure 2. Note that in Figure 2(a) all mass is concentrated around the center, since the prior mean is zero. Conversely, in Figure 2(b) all available space is evenly covered due to the uniform prior, resulting in more separable clusters in $\mathcal{S}^2$ compared to $\mathbb{R}^2$. However, as dimensionality increases, the Gaussian distribution starts to approximate a hypersphere, while its posterior becomes more expressive than the vMF due to the higher number of variance parameters. Simultaneously, as described in Subsection 3.5, the surface area of the vMF starts to collapse limiting the available space.

In Figure 7 and 8 of Appendix G, we present randomly generated samples from the $\mathcal{N}$-VAE and the $\mathcal{S}$-VAE, respectively. Moreover, in Figure 9 of Appendix G, we show 2-dimensional manifolds for the two models. Interestingly, the manifold given by the $\mathcal{S}$-VAE indeed results in a latent space where digits occupy the entire space and there is a sense of continuity from left to right.

5.3 SEMI-SUPERVISED LEARNING

Having observed the $\mathcal{S}$-VAE's ability to increase clusterability of data points in the latent space, we wish to further investigate this property using a semi-supervised classification task. For this purpose we re-implemented the M1 and M1+M2 models as described in (Kingma et al., 2014), and evaluate the classification accuracy of the $\mathcal{S}$-VAE and the $\mathcal{N}$-VAE on dynamically binarized MNIST. In the M1 model, a classifier utilizes the latent features obtained using a VAE as in experiment 5.2. The M1+M2 model is constructed by stacking the M2 model on top of M1, where M2 is the result of augmenting the VAE by introducing a partially observed variable $\mathbf{y}$, and combining the ELBO and classification objective. This concatenated model is trained end-to-end [3].

[3]It is worth noting that in the original implementation by Kingma et al. (2014) the stacked model did not converge well using end-to-end training, and used the extracted features of the



(a) $\mathbb{R}^2$ latent space of the $\mathcal{N}$-VGAE. (b) Hammer projection of $\mathcal{S}^2$ latent space of the $\mathcal{S}$-VGAE.

Figure 3: Latent space of unsupervised $\mathcal{N}$-VGAE and $\mathcal{S}$-VGAE models trained on Cora citation network. Colors denote documents classes which are not provided during training. (Best viewed in color)

This last model also allows for a combination of the two topologies due to the presence of two distinct latent variables, $\mathbf{z}_1$ and $\mathbf{z}_2$. Since in the M2 latent space the class assignment is expressed by the variable $\mathbf{y}$, while $\mathbf{z}_2$ only needs to capture the style, it naturally follows that the $\mathcal{N}$-VAE is more suited for this objective due to its higher number of variance parameters. Hence, besides comparing the $\mathcal{S}$-VAE against the $\mathcal{N}$-VAE, we additionally run experiments for the M1+M2 model by modeling $\mathbf{z}_1$, $\mathbf{z}_2$ respectively with a vMF and normal distribution.

Results As can be see in Table 2, for M1 the $\mathcal{S}$-VAE outperforms the $\mathcal{N}$-VAE in all dimensions up to $d = 40$. This result is amplified for a low number of observed labels. Note that for both models absolute performance drops as the dimensionality increases, since K-NN used as the classifier suffers from the curse of dimensionality. Besides reconfirming superiority of the $\mathcal{S}$-VAE in $d < 20$, its better performance than the $\mathcal{N}$-VAE for $d = 20$ was unexpected. This indicates that although the log-likelihood might be comparable(see Table 1) for higher dimensions, the $\mathcal{S}$-VAE latent space better captures the cluster structure.

In the concatenated model M1+M2, we first observe in Table 3 that either the pure $\mathcal{S}$-VAE or the $\mathcal{S}$+$\mathcal{N}$-VAE model yields the best results, where the $\mathcal{S}$-VAE almost always outperforms the $\mathcal{N}$-VAE. Our hypothesis regarding the merit of a $\mathcal{S}$+$\mathcal{N}$-VAE model is further confirmed, as displayed by the stable, strong performance across all different dimensions. Furthermore, the clear edge in clusterability of the $\mathcal{S}$-VAE in low dimensional $\mathbf{z}_1$ as already observed in Table 2, is again evident. As the dimensionality of $\mathbf{z}_1, \mathbf{z}_2$ increases, the accuracy of the $\mathcal{N}$-VAE improves, reducing the performance gap with the $\mathcal{S}$-VAE. As previously noticed the $\mathcal{S}$-VAE performance drops when $dim\ \mathbf{z}_2 = 50$, with the best result being obtained for $dim\ \mathbf{z}_1 = dim\ \mathbf{z}_2 = 10$. In fact, it is worth noting that for this setting the $\mathcal{S}$-VAE obtains comparable results to the original settings of (Kingma et al., 2014), while needing a considerably smaller latent space. Finally, the end-to-end trained $\mathcal{S}$+$\mathcal{N}$-VAE model is able to reach a significantly higher classification accuracy than the original results reported by Kingma et al. (2014), 96.7$_{\pm.1}$.

The M1+M2 model allows for conditional generation. Similarly to (Kingma et al., 2014), we set the latent variable $\mathbf{z}_2$ to the value inferred from the test image by the inference network, and then varied the class label $\mathbf{y}$. In Figure 10 of Appendix H we notice that the model is able to disentangle the style from the class.

Table 3: Summary of results of semi-supervised model M1+M2 on MNIST.

Method		**100**		
$dim\ \mathbf{z}_1$	$dim\ \mathbf{z}_2$	$\mathcal{N}$+$\mathcal{N}$	$\mathcal{S}$+$\mathcal{S}$	$\mathcal{S}$+$\mathcal{N}$
5	5	90.0$_{\pm.4}$	**94.0**$_{\pm.1}$	93.8$_{\pm.1}$
	10	90.7$_{\pm.3}$	94.1$_{\pm.1}$	**94.8**$_{\pm.2}$
	50	90.7$_{\pm.1}$	92.7$_{\pm.2}$	**93.0**$_{\pm.1}$
10	5	90.7$_{\pm.3}$	91.7$_{\pm.5}$	**94.0**$_{\pm.4}$
	10	92.2$_{\pm.1}$	**96.0**$_{\pm.2}$	**95.9**$_{\pm.3}$
	50	92.9$_{\pm.4}$	95.1$_{\pm.2}$	**95.7**$_{\pm.1}$
50	5	92.0$_{\pm.2}$	91.7$_{\pm.4}$	**95.8**$_{\pm.1}$
	10	93.0$_{\pm.1}$	95.8$_{\pm.1}$	**97.1**$_{\pm.1}$
	50	93.2$_{\pm.2}$	94.2$_{\pm.1}$	**97.4**$_{\pm.1}$

M1 model as inputs for the M2 model instead.

5.4 LINK PREDICTION ON GRAPHS

In this experiment, we aim at demonstrating the ability of the $\mathcal{S}$-VAE to learn meaningful embeddings of nodes in a graph, showing the advantages of embedding objects in a non-Euclidean space. We test hyperspherical reparameterization on the recently introduced Variational Graph Auto-Encoder (VGAE) (Kipf and Welling, 2016), a VAE model for graph-structured data. We perform training on a link prediction task on three popular citation network datasets (Sen et al., 2008): Cora, Citeseer and Pubmed.

Dataset statistics and further experimental details are summarized in Appendix F.3. The models are trained in an unsupervised fashion on a masked version of these datasets where some of the links have been removed. All node features are provided and efficacy is measured in terms of average precision (AP) and area under the ROC curve (AUC) on a test set of previously removed links. We use the same training, validation, and test splits as in Kipf and Welling (2016), i.e. we assign 5% of links for validation and 10% of links for testing.

Table 4: Results for link prediction in citation networks.

Method		$\mathcal{N}$**-VGAE**	$\mathcal{S}$**-VGAE**
Cora	AUC	$92.7_{\pm.2}$	$\mathbf{94.1}_{\pm.1}$
	AP	$93.2_{\pm.4}$	$\mathbf{94.1}_{\pm.3}$
Citeseer	AUC	$90.3_{\pm.5}$	$\mathbf{94.7}_{\pm.2}$
	AP	$91.5_{\pm.5}$	$\mathbf{95.2}_{\pm.2}$
Pubmed	AUC	$\mathbf{97.1}_{\pm.0}$	$96.0_{\pm.1}$
	AP	$\mathbf{97.1}_{\pm.0}$	$96.0_{\pm.1}$

Results In Table 4, we show that our model outperforms the $\mathcal{N}$-VGAE baseline on two out of the three datasets by a significant margin. The log-probability of a link is computed as the dot product of two embeddings. In a hypersphere, this can be interpreted as the cosine similarity between vectors. Indeed we find that the choice of a dot product scoring function for link prediction is problematic in combination with the normal distribution on the latent space. If embeddings are close to the zero-center, noise during training can have a large destabilizing effect on the angle information between two embeddings. In practice, the model finds a solution where embeddings are "pushed" away from the zero-center, as demonstrated in Figure 3(a). This counteracts the pull towards the center arising from the standard prior and can overall lead to poor modeling performance. By constraining the embeddings to the surface of a hypersphere, this effect is mitigated, and the model can find a good separation of the latent clusters, as shown in Figure 3(b).

On Pubmed, we observe that the $\mathcal{S}$-VAE converges to a lower score than the $\mathcal{N}$-VAE. The Pubmed dataset is significantly larger than Cora and Citeseer, and hence more complex. The $\mathcal{N}$-VAE has a larger number of variance parameters for the posterior distribution, which might have played an important role in better modeling the relationships between nodes. We further hypothesize that not all graphs are necessarily better embedded in a hyperspherical space and that this depends on some fundamental topological properties of the graph. For instance, the already mentioned work from Nickel and Kiela (2017) shows that hyperbolical space is better suited for graphs with a hierarchical, tree-like structure. These considerations prefigure an interesting research direction that will be explored in future work.

6 CONCLUSION

With the $\mathcal{S}$-VAE we set an important first step in the exploration of hyperspherical latent representations for variational auto-encoders. Through various experiments, we have shown that $\mathcal{S}$-VAEs have a clear advantage over $\mathcal{N}$-VAEs for data residing on a known hyperspherical manifold, and are competitive or surpass $\mathcal{N}$-VAEs for data with a non-obvious hyperspherical latent representation in lower dimensions. Specifically, we demonstrated $\mathcal{S}$-VAEs improve separability in semi-supervised classification and that they are able to improve results on state-of-the-art link prediction models on citation graphs, by merely changing the prior and posterior distributions as a simple drop-in replacement.

We believe that the presented research paves the way for various promising areas of future work, such as exploring more flexible approximate posterior distributions through normalizing flows on the hypersphere, or hierarchical mixture models combining hyperspherical and hyperplanar space. Further research should be done in increasing the performance of $\mathcal{S}$-VAEs in higher dimensions; one possible solution of which could be to dynamically learn the radius of the latent hypersphere in a full Bayesian setting.

Acknowledgements

We would like to thank Rianne van den Berg, Jonas Köhler, Pim de Haan, Taco Cohen, Marco Federici, and Max Welling for insightful discussions. T.K. is supported by the SAP Innovation Center Network. J.M.T. was funded by the European Commission within the Marie Skłodowska-Curie Individual Fellowship (Grant No. 702666, Deep learning and Bayesian inference for medical imaging).

References

Aytekin, C., Ni, X., Cricri, F., and Aksu, E. (2018). Clustering and unsupervised anomaly detection with l2 normalized deep auto-encoder representations. *arXiv preprint arXiv:1802.00187*.

Banerjee, A. and Basu, S. (2007). Topic models over text streams: A study of batch and online unsupervised learning. *ICDM*, pages 431–436.

Banerjee, A., Dhillon, I., Ghosh, J., and Sra, S. (2003). Generative model-based clustering of directional data. *SIGKDD*, pages 19–28.

Banerjee, A., Dhillon, I. S., Ghosh, J., and Sra, S. (2005). Clustering on the unit hypersphere using von mises-fisher distributions. *Journal of Machine Learning Research*, 6(Sep):1345–1382.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A. M., Jozefowicz, R., and Bengio, S. (2015). Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*.

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

Cohen, T. S., Geiger, M., Khler, J., and Welling, M. (2018). Spherical CNNs. *ICLR*.

Dilokthanakul, N., Mediano, P. A. M., Garnelo, M., Lee, M. C. H., Salimbeni, H., Arulkumaran, K., and Shanahan, M. (2016). Deep unsupervised clustering with gaussian mixture variational autoencoders. *CoRR*, abs/1611.02648.

Fisher, N. I., Lewis, T., and Embleton, B. J. (1987). *Statistical analysis of spherical data*. Cambridge university press.

Glorot, X. and Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. *AISTATS*, pages 249–256.

Gopal, S. and Yang, Y. (2014). Von mises-fisher clustering models. *ICML*, pages 154–162.

Guan, H. and Smith, W. A. (2017). Structure-from-motion in spherical video using the von mises-fisher distribution. *IEEE Transactions on Image Processing*, 26(2):711–723.

Guu, K., Hashimoto, T. B., Oren, Y., and Liang, P. (2017). Generating sentences by editing prototypes. *arXiv preprint arXiv:1709.08878*.

Hasnat, M., Bohné, J., Milgram, J., Gentric, S., Chen, L., et al. (2017). von mises-fisher mixture model-based deep learning: Application to face verification. *arXiv preprint arXiv:1706.04264*.

Kingma, D. and Welling, M. (2014). Efficient gradient-based inference through transformations between bayes nets and neural nets. *ICML*, pages 1782–1790.

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

Kingma, D. P., Mohamed, S., Rezende, D. J., and Welling, M. (2014). Semi-supervised learning with deep generative models. *NIPS*, pages 3581–3589.

Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. *CoRR*, abs/1312.6114.

Kipf, T. N. and Welling, M. (2016). Variational Graph Auto-Encoders. *NIPS Bayesian Deep Learning Workshop*.

Liu, C. and Zhu, J. (2017). Riemannian Stein Variational Gradient Descent for Bayesian Inference. *ArXiv e-prints*.

Liu, W., Zhang, Y.-M., Li, X., Yu, Z., Dai, B., Zhao, T., and Song, L. (2017). Deep hyperspherical learning. *NIPS*, pages 3953–3963.

Mardia, K. V. (1975). Statistics of directional data. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 349–393.

Naesseth, C., Ruiz, F., Linderman, S., and Blei, D. (2017). Reparameterization Gradients through Acceptance-Rejection Sampling Algorithms. *AISTATS*, pages 489–498.

Nalisnick, E. and Smyth, P. (2017). Stick-breaking variational autoencoders. *ICLR*.

Nickel, M. and Kiela, D. (2017). Poincaré embeddings for learning hierarchical representations. *NIPS*, pages 6341–6350.

Reisinger, J., Waters, A., Silverthorn, B., and Mooney, R. J. (2010). Spherical topic models. *ICML*.

Rezende, D. and Mohamed, S. (2015). Variational inference with normalizing flows. *ICML*, 37:1530–1538.

Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*.

Salakhutdinov, R. and Murray, I. (2008). On the quantitative analysis of deep belief networks. *ICML*, pages 872–879.

Sen, P., Namata, G., Bilgic, M., Getoor, L., Galligher, B., and Eliassi-Rad, T. (2008). Collective classification in network data. *AI magazine*, 29(3):93.

Tomczak, J. M. and Welling, M. (2017). VAE with a VampPrior. *arXiv preprint arXiv:1705.07120*.

Ulrich, G. (1984). Computer generation of distributions on the m-sphere. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 33(2):158–163.

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

A SAMPLING PROCEDURE

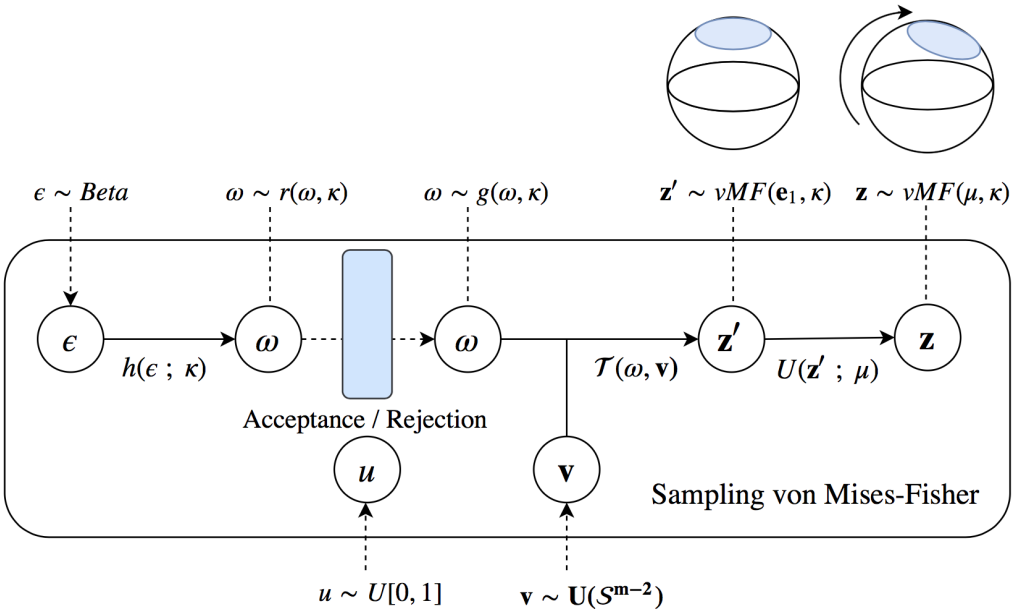


Figure 4: Overview of von Mises-Fisher sampling procedure. Note that as ω is a scalar, the procedure does not suffer from the curse of dimensionality.

The general algorithm for sampling from a vMF has been outlined in Algorithm 1. The exact form of the distribution of the univariate distribution $g(\omega|k)$ is:

$$g(\omega|k) = \frac{2(\pi^{m/2})}{\Gamma(m/2)} \mathcal{C}_m(k) \frac{\exp(\omega k)(1-\omega^2)^{\frac{1}{2}(m-3)}}{B(\frac{1}{2}, \frac{1}{2}(m-1))}, \tag{11}$$

Samples from this distribution are drawn using an acceptance/rejection algorithm when $m \neq 3$. The complete procedure is described in Algorithm 2. The $Householder$ reflection (see Algorithm 3 for details) simply finds an orthonormal transformation that maps the modal vector $\mathbf{e}_1 = (1, 0, \cdots, 0)$ to μ. Since an orthonormal transformation preserves the distances all the points in the hypersphere will stay in the surface after mapping. Notice that even the transform $U\mathbf{z}' = (\mathbb{I} - 2\mathbf{u}\mathbf{u}^\top)\mathbf{z}'$, can be executed in $\mathcal{O}(m)$ by rearranging the terms.

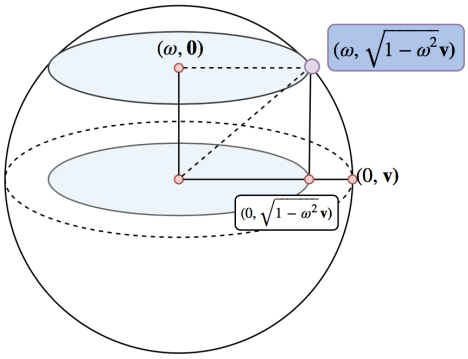


Figure 5: Geometric representation of a single sample in $\mathcal{S}^2$, where $\omega \sim g(\omega|k)$ and $\mathbf{v} \sim U(\mathcal{S}^1)$.

Algorithm 2 $g(\omega|k)$ acceptance-rejection sampling

Input: dimension m, concentration κ
Initialize values:
$b \leftarrow \dfrac{-2k + \sqrt{4k^2 + (m-1)^2}}{m-1}$
$a \leftarrow \dfrac{(m-1) + 2k + \sqrt{4k^2 + (m-1)^2}}{4}$
$d \leftarrow \dfrac{4ab}{(1+b)} - (m-1)\ln(m-1)$
repeat
  Sample $\varepsilon \sim$ Beta$(\frac{1}{2}(m-1), \frac{1}{2}(m-1))$
  $\omega \leftarrow h(\varepsilon, k) = \dfrac{1-(1+b)\varepsilon}{1-(1-b)\varepsilon}$
  $t \leftarrow \dfrac{2ab}{1-(1-b)\varepsilon}$
  Sample $u \sim \mathcal{U}(0,1)$
until $(m-1)\ln(t) - t + d \geq \ln(u)$
Return: ω

Algorithm 3 Householder transform

Input: mean μ, modal vector $\mathbf{e}_1$
$\mathbf{u}' \leftarrow \mathbf{e}_1 - \mu$
$\mathbf{u} \leftarrow \frac{\mathbf{u}'}{||\mathbf{u}'||}$
$U \leftarrow \mathbb{I} - 2\mathbf{u}\mathbf{u}^\top$
Return: U

Table 5: Expected number of samples needed before acceptance, computed using Monte Carlo estimate with 1000 samples varying dimensionality and concentration parameters. Notice that the sampling complexity increases in κ, but decreases as the dimensionality, d, increases.

	$\kappa=1$	$\kappa=5$	$\kappa=10$	$\kappa=50$	$\kappa=100$	$\kappa=500$	$\kappa=1000$	$\kappa=5000$	$\kappa=10000$
$d=5$	1.020	1.171	1.268	1.398	1.397	1.426	1.458	1.416	1.440
$d=10$	1.008	1.094	1.154	1.352	1.411	1.407	1.369	1.402	1.419
$d=20$	1.001	1.031	1.085	1.305	1.342	1.367	1.409	1.410	1.407
$d=40$	1.000	1.011	1.027	1.187	1.288	1.397	1.433	1.402	1.423
$d=100$	1.000	1.000	1.006	1.092	1.163	1.317	1.360	1.398	1.416

B KL DIVERGENCE DERIVATION

The KL divergence between a von-Mises-Fisher distribution $q(\mathbf{z}|\mu,k)$ and an uniform distribution in the hypersphere (one divided by the surface area of $\mathcal{S}^{m-1}$) $p(\mathbf{x}) = \left(\dfrac{2(\pi^{m/2})}{\Gamma(m/2)}\right)^{-1}$ is:

$$\mathcal{KL}[q(\mathbf{z}|\mu,k) \;||\; p(\mathbf{z})] = \int_{\mathcal{S}^{m-1}} q(\mathbf{z}|\mu,k)\log\frac{q(\mathbf{z}|\mu,k)}{p(\mathbf{z})}d\mathbf{z} \tag{12}$$

$$= \int_{\mathcal{S}^{m-1}} q(\mathbf{z}|\mu,k)\left(\log\mathcal{C}_m(k) + k\mu^T\mathbf{z} - \log p(\mathbf{z})\right)d\mathbf{z} \tag{13}$$

$$= k\mu\,\mathbb{E}_q[\mathbf{z}] + \log\mathcal{C}_m(k) - \log\left(\frac{2(\pi^{m/2})}{\Gamma(m/2)}\right)^{-1} \tag{14}$$

$$= k\frac{\mathcal{I}_{m/2}(k)}{\mathcal{I}_{m/2-1}(k)} + \left((m/2-1)\log k - (m/2)\log(2\pi) - \log\mathcal{I}_{m/2-1}(k)\right) \tag{15}$$

$$+ \frac{m}{2}\log\pi + \log 2 - \log\Gamma(\frac{m}{2}),$$

B.1 GRADIENT OF KL DIVERGENCE

Using

$$\nabla_k \mathcal{I}_v(k) = \frac{1}{2}\left(\mathcal{I}_{v-1}(k) + \mathcal{I}_{v+1}(k)\right), \tag{16}$$

and

$$\nabla_k \log \mathcal{C}_m(k) = \nabla_k \left((m/2-1)\log k - (m/2)\log(2\pi) - \log \mathcal{I}_{m/2-1}(k)\right) \tag{17}$$

$$= -\frac{\mathcal{I}_{m/2}(k)}{\mathcal{I}_{m/2-1}(k)}, \tag{18}$$

then

$$\nabla_\kappa \mathcal{KL}[q(\mathbf{z}|\mu,k) \mid\mid p(\mathbf{z})] = \nabla_\kappa k \frac{\mathcal{I}_{m/2}(k)}{\mathcal{I}_{m/2-1}(k)} + \nabla_k \log \mathcal{C}_m(k) \tag{19}$$

$$= \frac{\mathcal{I}_{m/2}(k)}{\mathcal{I}_{m/2-1}(k)} + k\nabla_k \frac{\mathcal{I}_{m/2}(k)}{\mathcal{I}_{m/2-1}(k)} - \frac{\mathcal{I}_{m/2}(k)}{\mathcal{I}_{m/2-1}(k)} \tag{20}$$

$$= \frac{1}{2}k\left(\frac{\mathcal{I}_{m/2+1}(k)}{\mathcal{I}_{m/2-1}(k)} - \frac{\mathcal{I}_{m/2}(k)\left(\mathcal{I}_{m/2-2}(k) + \mathcal{I}_{m/2}(k)\right)}{\mathcal{I}_{m/2-1}(k)^2} + 1\right), \tag{21}$$

Notice that we can use $\mathcal{I}^{exp}_{m/2} = \exp(-k)\mathcal{I}_{m/2}$ for numerical stability.

C PROOF OF LEMMA 2

Lemma 3 (2). *Let f be any measurable function and $\varepsilon \sim \pi_1(\varepsilon|\theta) = s(\varepsilon)\frac{g(h(\varepsilon,\theta)|\theta)}{r(h(\varepsilon,\theta)|\theta)}$ the distribution of the accepted sample. Also let $\mathbf{v} \sim \pi_2(\mathbf{v})$, and $\mathcal{T}$ a transformation that depends on the parameters such that if $\mathbf{z} = \mathcal{T}(\omega, \mathbf{v}; \theta)$ with $\omega \sim g(\omega|\theta)$, then $\mathbf{z} \sim q(\mathbf{z}|\theta)$:*

$$\mathbb{E}_{(\varepsilon,\mathbf{v})\sim\pi_1(\varepsilon|\theta)\pi_2(\mathbf{v})}\left[f\left(\mathcal{T}(h(\varepsilon,\theta),\mathbf{v};\theta)\right)\right] = \int f(\mathbf{z})q(\mathbf{z}|\theta)d\mathbf{z} = \mathbb{E}_{q(\mathbf{z}|\theta)}[f(\mathbf{z})], \tag{22}$$

Proof.

$$\mathbb{E}_{(\varepsilon,\mathbf{v})\sim\pi_1(\varepsilon|\theta)\pi_2(\mathbf{v})}\left[f\left(\mathcal{T}(h(\varepsilon,\theta),\mathbf{v};\theta)\right)\right] = \iint f\left(\mathcal{T}(h(\varepsilon,\theta),\mathbf{v};\theta)\right)\pi_1(\varepsilon|\theta)\pi_2(\mathbf{v})d\varepsilon d\mathbf{v}, \tag{23}$$

Using the same argument employed by Naesseth et al. (2017) we can apply the change of variables $\omega = h(\varepsilon,\theta)$ rewrite the expression as:

$$= \iint f\left(\mathcal{T}(\omega,\mathbf{v};\theta)\right) g(\omega|\theta)\pi_2(\mathbf{v})d\omega d\mathbf{v} =^* \int f(\mathbf{z})q(\mathbf{z}|\theta)d\mathbf{z} \tag{24}$$

Where in * we applied the change of variables $\mathbf{z} = \mathcal{T}(\omega,\mathbf{v};\theta)$. □

D REPARAMETRIZATION GRADIENT DERIVATION

D.1 GENERAL EXPRESSION DERIVATION

We can then proceed as in 8 using Lemma 2 and the the log derivative trick to compute the gradient of the expectation term $\nabla_\theta \mathbb{E}_{q(\mathbf{z}|\theta)}[f(\mathbf{z})]$:

$$\nabla_\theta \mathbb{E}_{q(\mathbf{z}|\theta)}[f(\mathbf{z})] = \nabla_\theta \iint f\left(\mathcal{T}(h(\varepsilon,\theta),\mathbf{v};\theta)\right)\pi_1(\varepsilon|\theta)\pi_2(\mathbf{v})d\varepsilon d\mathbf{v} \tag{25}$$

$$= \nabla_\theta \iint f\left(\mathcal{T}(h(\varepsilon,\theta),\mathbf{v};\theta)\right) s(\varepsilon) \frac{g(h(\varepsilon,\theta)|\theta)}{r(h(\varepsilon,\theta)|\theta)} \pi_2(\mathbf{v}) d\varepsilon d\mathbf{v} \tag{26}$$

$$= \iint s(\varepsilon)\pi_2(\mathbf{v}) \nabla_\theta \left(f\left(\mathcal{T}(h(\varepsilon,\theta),\mathbf{v};\theta)\right) \frac{g(h(\varepsilon,\theta)|\theta)}{r(h(\varepsilon,\theta)|\theta)} \right) d\varepsilon d\mathbf{v} \tag{27}$$

$$= \iint s(\varepsilon)\pi_2(\mathbf{v}) \frac{g(h(\varepsilon,\theta)|\theta)}{r(h(\varepsilon,\theta)|\theta)} \nabla_\theta \left(f\left(\mathcal{T}(h(\varepsilon,\theta),\mathbf{v};\theta)\right)\right) d\varepsilon d\mathbf{v} \tag{28}$$

$$+ \iint s(\varepsilon)\pi_2(\mathbf{v}) f\left(\mathcal{T}(h(\varepsilon,\theta),\mathbf{v};\theta)\right) \nabla_\theta \left(\frac{g(h(\varepsilon,\theta)|\theta)}{r(h(\varepsilon,\theta)|\theta)} \right) d\varepsilon d\mathbf{v}$$

$$= \iint \pi_1(\varepsilon|\theta)\pi_2(\mathbf{v}) \nabla_\theta \left(f\left(\mathcal{T}(h(\varepsilon,\theta),\mathbf{v};\theta)\right)\right) d\varepsilon dv \tag{29}$$

$$+ \iint s(\varepsilon)\pi_2(\mathbf{v}) f\left(\mathcal{T}(h(\varepsilon,\theta),\mathbf{v};\theta)\right) \nabla_\theta \left(\frac{g(h(\varepsilon,\theta)|\theta)}{r(h(\varepsilon,\theta)|\theta)} \right) d\varepsilon d\mathbf{v}$$

$$= \underbrace{\mathbb{E}_{(\varepsilon,\mathbf{v})\sim\pi_1(\varepsilon|\theta)\pi_2(\mathbf{v})} \left[\nabla_\theta f\left(\mathcal{T}(h(\varepsilon,\theta),\mathbf{v};\theta)\right)\right]}_{g_{rep}} \tag{30}$$

$$+ \underbrace{\mathbb{E}_{(\varepsilon,\mathbf{v})\sim\pi_1(\varepsilon|\theta)\pi_2(\mathbf{v})} \left[f\left(\mathcal{T}(h(\varepsilon,\theta),\mathbf{v};\theta)\right) \nabla_\theta \log \left(\frac{g(h(\varepsilon,\theta)|\theta)}{r(h(\varepsilon,\theta)|\theta)} \right) \right]}_{g_{cor}},$$

where g_{rep} is the reparameterization term and g_{cor} the correction term. Since h is invertible in ε, Naesseth et al. (2017) show that $\nabla_\theta \log \frac{q(h(\varepsilon,\theta),\theta)}{r((h(\varepsilon,\theta),\theta)}$ in g_{cor} simplifies to:

$$\nabla_\theta \log \frac{g(h(\varepsilon,\theta),\theta)}{r((h(\varepsilon,\theta),\theta)} = \nabla_\theta \log g(h(\varepsilon,\theta),\theta) + \nabla_\theta \log |\frac{\partial h(\varepsilon,\theta)}{\partial \varepsilon}|, \tag{31}$$

D.2 GRADIENT CALCULATION

In our specific case we want to take the gradient w.r.t. θ of the expression:

$$\mathbb{E}_{q_\psi(\mathbf{z}|\mathbf{x};\theta)}[\log p_\phi(\mathbf{x}|\mathbf{z})] \quad \text{where } \theta = (\mu, \kappa), \tag{32}$$

The gradient can be computed using the Lemma 2 and the subsequent gradient derivation with $f(\mathbf{z}) = p_\phi(\mathbf{x}|\mathbf{z})$. As specified in Section 3.4 we optimize unbiased Monte Carlo estimates of the gradient. Therefore fixed one datapoint $\mathbf{x}$ and sampled $(\varepsilon, \mathbf{v}) \sim \pi_1(\varepsilon|\theta)\pi_2(\mathbf{v})$ the gradient is:

$$\nabla_\theta \mathbb{E}_{q_\psi(\mathbf{z}|\mathbf{x};\theta)}[\log p_\phi(\mathbf{x}|\mathbf{z})] = g_{rep} + g_{cor}, \tag{33}$$

With

$$g_{rep} \approx \nabla_\theta \log p_\phi\left(\mathbf{x}|\mathcal{T}(h(\varepsilon,\theta),\mathbf{v};\theta)\right), \tag{34}$$

$$g_{cor} \approx p_\phi\left(x|\mathcal{T}(h(\varepsilon,\theta),\mathbf{v};\theta)\right) \left(\nabla_\theta \log g(h(\varepsilon,\theta)|\theta) + \nabla_\theta \log |\frac{\partial h(\varepsilon,\theta)}{\partial \varepsilon}| \right), \tag{35}$$

where g_{rep} is simply the gradient of the reconstruction loss w.r.t θ and can be easily handled by automatic differentiation packages.

For what concerns g_{cor} we notice that the terms $g()$ and $h()$ do not depend on μ. Thus the g_{cor} term w.r.t. μ is 0 an all the following calculations can will be only w.r.t. κ. We therefore have:

$$\frac{\partial h(\varepsilon,k)}{\partial \varepsilon} = \frac{-2b}{((b-1)\varepsilon+1)^2} \quad \text{where } b = \frac{-2k + \sqrt{4k^2 + (m-1)^2}}{m-1}, \tag{36}$$

and

$$\nabla_\kappa \log g(\omega|k) = \nabla_\kappa \left(\log \mathcal{C}_m(k) + \omega k + \frac{1}{2}(m-3)\log(1-\omega^2) \right) \tag{37}$$

$$= \nabla_k \log \mathcal{C}_m(k) + \nabla_\kappa \left(\omega k + \frac{1}{2}(m-3)\log(1-\omega^2) \right). \tag{38}$$

So, putting everything together we have:

$$g_{cor} = \log p_\phi(x|z) \cdot \left[-\frac{\mathcal{I}_{m/2}}{\mathcal{I}_{m/2-1}} + \nabla_\kappa \left(\omega\kappa + \frac{1}{2}(m-3)\log(1-\omega^2) + \log|\frac{-2b}{((b-1)\varepsilon+1)^2}| \right) \right], \tag{39}$$

where

$$b = \frac{-2k + \sqrt{4k^2 + (m-1)^2}}{m-1} \tag{40}$$

$$\omega = h(\varepsilon, \theta) = \frac{1-(1+b)\varepsilon}{1-(1-b)\varepsilon} \tag{41}$$

$$z = \mathcal{T}(h(\varepsilon, \theta), \mathbf{v}; \theta), \tag{42}$$

And the term $\nabla_\kappa \left(\omega\kappa + \frac{1}{2}(m-3)\log(1-\omega^2) + \log|\frac{-2b}{((b-1)\varepsilon+1)^2}| \right)$ can be computed by automatic differentiation packages.

E COLLAPSE OF THE SURFACE AREA

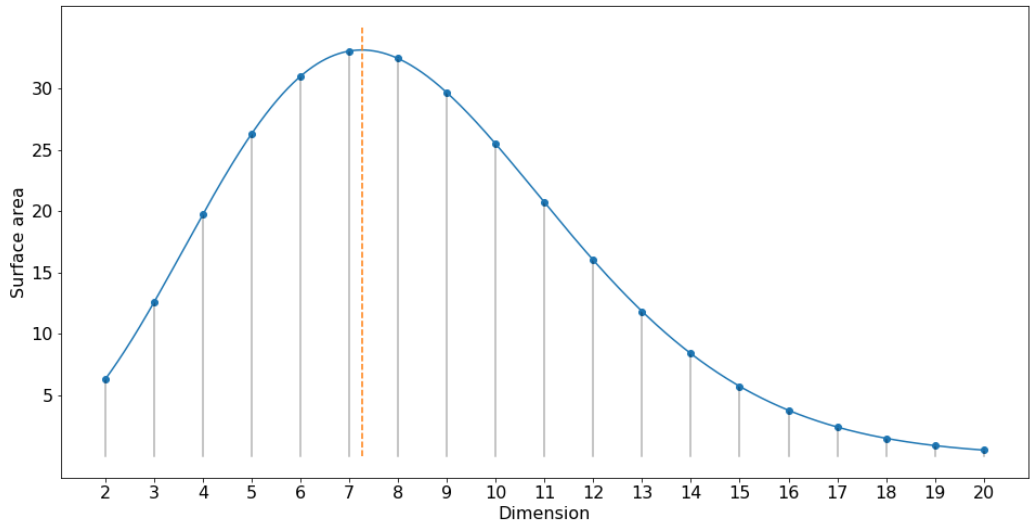


Figure 6: Plot of the unit hyperspherical surface area against dimensionality. The surface area has a maximum for $m = 7$.

F EXPERIMENTAL DETAILS: ARCHITECTURE AND HYPERPARAMETERS

F.1 EXPERIMENT 5.2

Architecture and hyperparameters For both the encoder and the decoder we use MLPs with 2 hidden layers of respectively, [256, 128] and [128, 256] hidden units. We trained until convergence using early-stopping with a look

ahead of 50 epochs. We used the Adam optimizer (Kingma and Ba, 2014) with a learning rate of 1e-3, and mini-batches of size 64. Additionally, we used a linear *warm-up* for 100 epochs (Bowman et al., 2015). The weights of the neural network were initialized according to (Glorot and Bengio, 2010).

F.2 EXPERIMENT 5.3

Architecture and Hyperparameters For M1 we reused the trained models of the previous experiment, and used K-nearest neighbors (K-NN) as a classifier with $k = 5$. In the $\mathcal{N}$-VAE case we used the Euclidean distance as a distance metric. For the $\mathcal{S}$-VAE the geodesic distance $\arccos(\mathbf{x}^\top \mathbf{y})$ was employed. The performance was evaluated for $N = [100, 600, 1000]$ observed labels.

The stacked M1+M2 model uses the same architecture as outlined by Kingma et al. (2014), where the MLPs utilized in the generative and inference models are constructed using a single hidden layer, each with 500 hidden units. The latent space dimensionality of $\mathbf{z}_1$, $\mathbf{z}_2$ were both varied in $[5, 10, 50]$. We used the rectified linear unit (ReLU) as an activation function. Training was continued until convergence using early-stopping with a look ahead of 50 epochs on the validation set. We used the Adam optimizer with a learning rate of 1e-3, and mini-batches of size 100. All neural network weight were initialized according to (Glorot and Bengio, 2010). N was set to 100, and the α parameter used to scale the classification loss was chosen between $[0.1, 1.0]$. Crucially, we train this model end-to-end instead of by parts.

F.3 EXPERIMENT 5.4

Architecture and Hyperparameters We are training a Variational Graph Auto-encoder (VGAE) model, a state-of-the-art link prediction model for graphs, as proposed in Kipf and Welling (2016). For a fair comparison, we use the same architecture as in the original paper and we just change the way the latent space is generated using the vMF distribution instead of a normal distribution. All models are trained for 200 epochs on Cora and Citeseer, and 400 epochs on Pubmed with the Adam optimizer. Optimal learning rate $lr \in \{0.01, 0.005, 0.001\}$, dropout rate $p_{do} \in \{0, 0.1, 0.2, 0.3, 0.4\}$ and number of latent dimensions $d_z \in \{8, 16, 32, 64\}$ are determined via grid search based on validation AUC performance. For $\mathcal{S}$-VGAE, we omit the $d_z = 64$ setting as some of our experiments ran out of memory. The model is trained with a single hidden layer with 32 units and with document features as input, as in Kipf and Welling (2016). The weights of the neural network were initialized according to (Glorot and Bengio, 2010). For testing, we report performance of the model selected from the training epoch with highest AUC score on the validation set. Different from (Kipf and Welling, 2016), we train both the $\mathcal{N}$-VGAE and the $\mathcal{S}$-VGAE models using negative sampling in order to speed up training, i.e. for each positive link we sample, uniformly at random, one negative link during every training epoch. All experiments are repeated 5 times, and we report mean and standard error values.

F.3.1 FURTHER EXPERIMENTAL DETAILS

Dataset statistics are summarized in Table 6. Final hyperparameter choices found via grid search on the validation splits are summarized in Table 7.

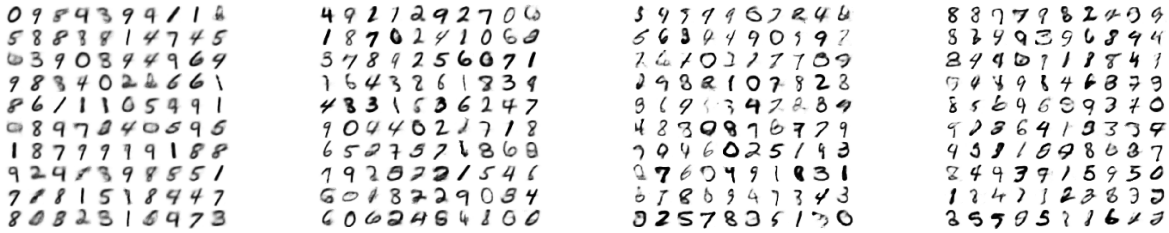
Table 6: Dataset statistics for citation network datasets.

Dataset	Nodes	Edges	Features
Cora	2,708	5,429	1,433
Citeseer	3,327	4,732	3,703
Pubmed	19,717	44,338	500

Table 7: Best hyperparameter settings found for citation network datasets.

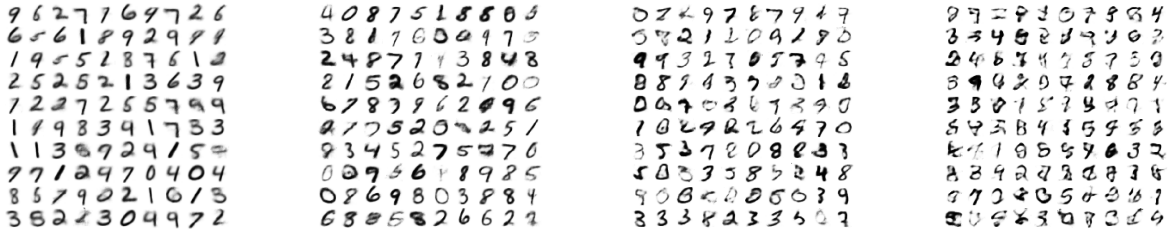
Dataset	Model	lr	p_{do}	d_z
Cora	$\mathcal{N}$-VAE	0.005	0.4	64
	$\mathcal{S}$-VAE	0.001	0.1	32
Citeseer	$\mathcal{N}$-VAE	0.01	0.4	64
	$\mathcal{S}$-VAE	0.005	0.2	32
Pubmed	$\mathcal{N}$-VAE	0.001	0.2	32
	$\mathcal{S}$-VAE	0.01	0.0	32

G VISUALIZATION OF SAMPLES AND LATENT SPACES



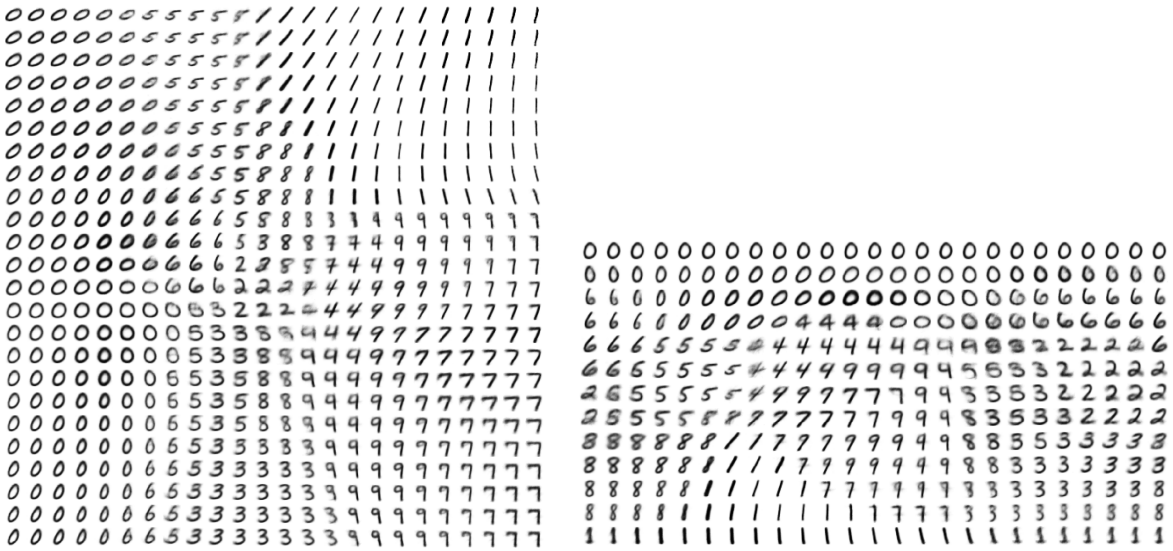
(a) $d = 2$ (b) $d = 5$ (c) $d = 10$ (d) $d = 20$

Figure 7: Random samples from $\mathcal{N}$-VAE of MNIST for different dimensionalities of latent space.



(a) $d = 2$ (b) $d = 5$ (c) $d = 10$ (d) $d = 20$

Figure 8: Random samples from $\mathcal{S}$-VAE of MNIST for different dimensionalities of latent space.



(a) $\mathcal{N}$-VAE (b) $\mathcal{S}$-VAE

Figure 9: Visualization of the 2 dimensional manifold of MNIST for both the $\mathcal{N}$-VAE and $\mathcal{S}$-VAE. Notice that the $\mathcal{N}$-VAE has a clear center and all digits are spread around it. Conversely, in the $\mathcal{S}$-VAE instead all digits occupy the entire space and there is a sense of continuity from left to right.

H VISUALIZATION OF CONDITIONAL GENERATION

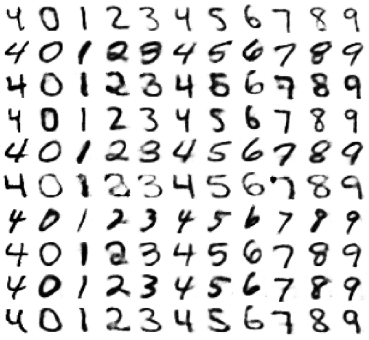


Figure 10: Visualization of handwriting styles learned by the model, using conditional generation on MNIST of M1+M2 with $dim(\mathbf{z}_1) = 50$, $dim(\mathbf{z}_2) = 50$, $\mathcal{S}$+$\mathcal{N}$. Following Kingma et al. (2014), the left most column shows images from the test set. The other columns show analogical fantasies of $\mathbf{x}$ by the generative model, where in each row the latent variable $\mathbf{z}_2$ is set to the value inferred from the test image by the inference network and the class label $\mathbf{y}$ is varied per column.

View publication stats