Learning Hyperbolic Representations for Unsupervised 3D Segmentation

Joy Hsu
@joycjhsu
joycj@stanford.edu

Overview

Segmenting coronavirus

Applications in cryo-EM

Hyperbolic representation learning

Segmenting coronavirus

Problem introduction

What do you do with an unknown and unexplored input?

How do you discover features of interest from it?

Object discovery

Object discovery

Unsupervised 3D segmentation

Voxel-level class prediction

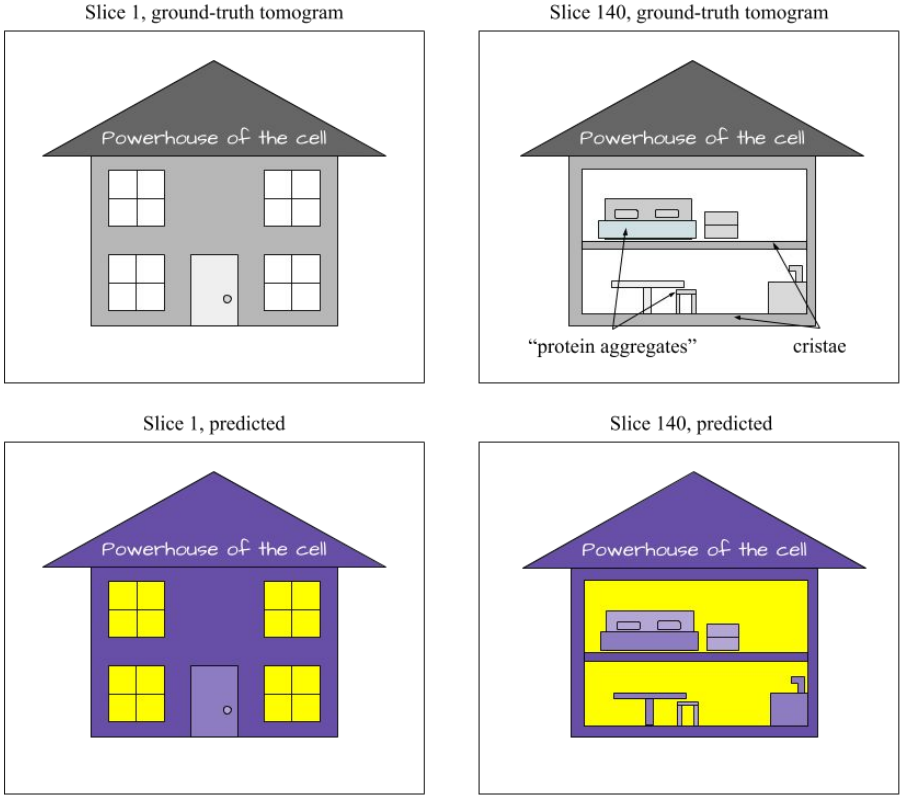
Focus on biomedical data

Object discovery

Unsupervised 3D segmentation

Voxel-level class prediction

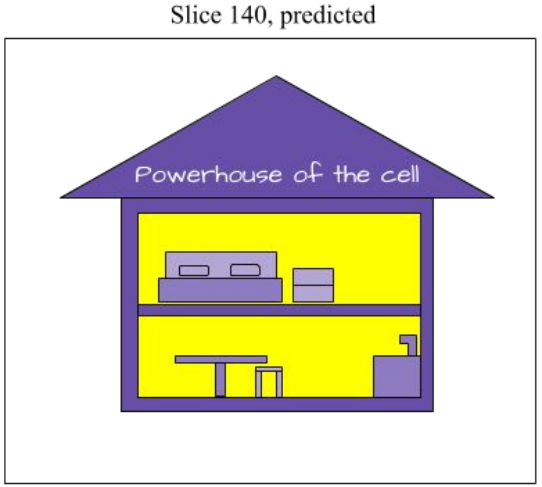
Focus on biomedical data



Granularity of segmentation

Many “correct” segmentations at different levels of semantic hierarchy

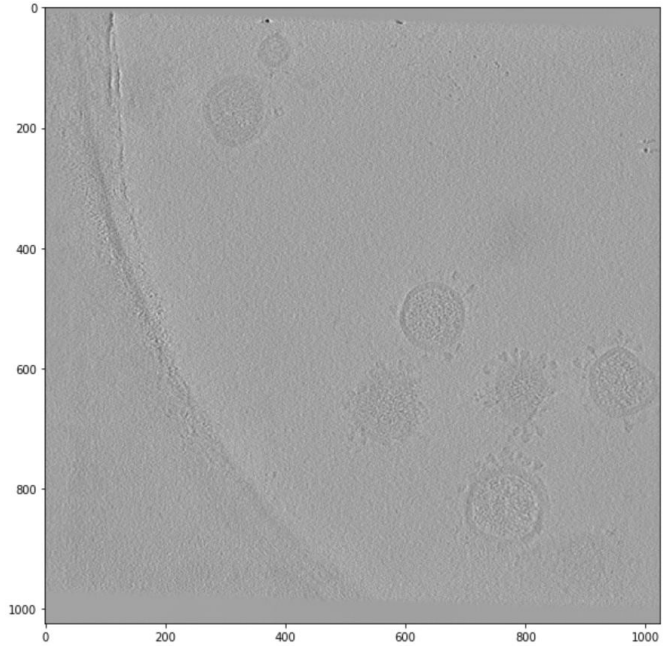
Difficult to do unsupervised segmentation without base granularity



Coronavirus scans

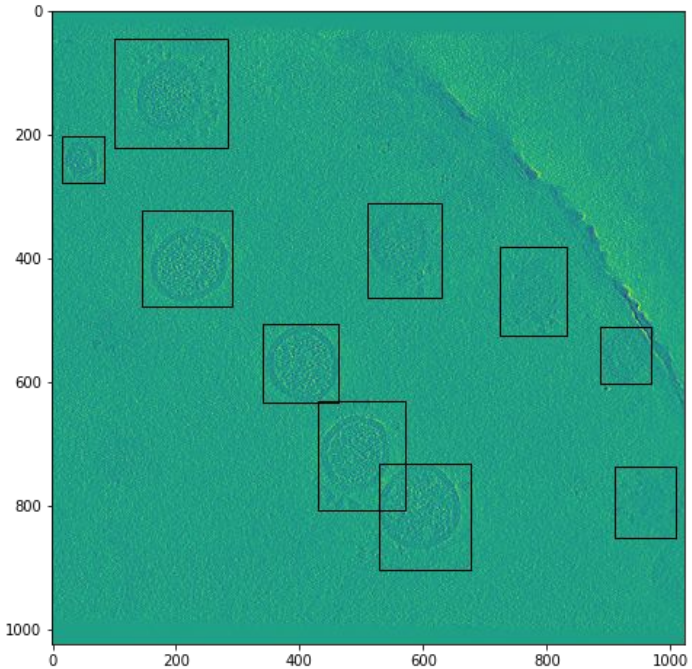
Slice of 3D tomogram

How do we learn more about spikes on coronavirus particles?

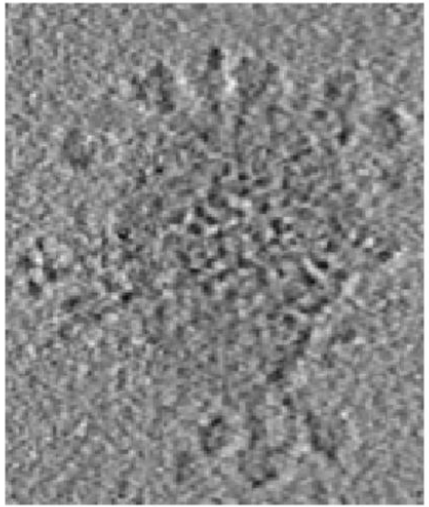


Automated particle picking

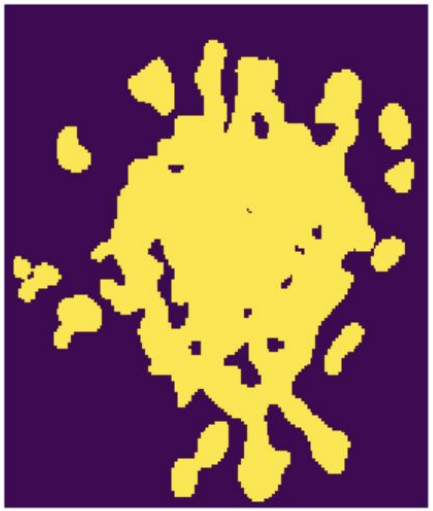
RetinaNet to identify particles



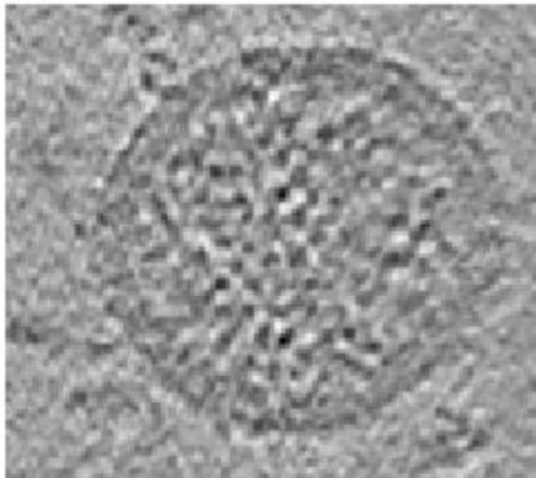
Spike discovery

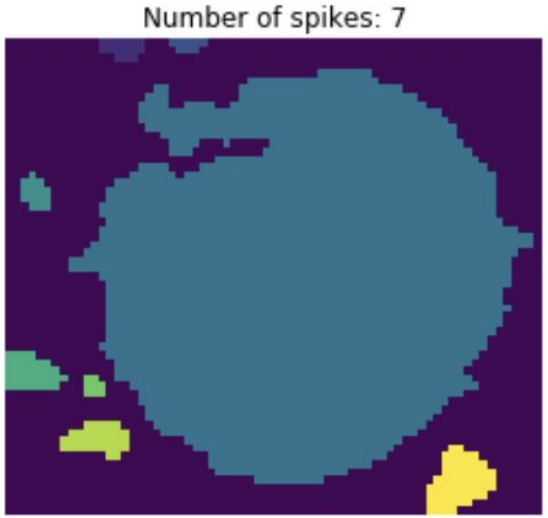


Our Model



Count spikes





Applications in Cryo-EM

CryoEM

Cryogenic electron microscopy

Allows for observation of biological specimens in their native environment at cryogenic temperatures

Produces high-resolution three-dimensional views of samples

Each voxel represents number of electrons

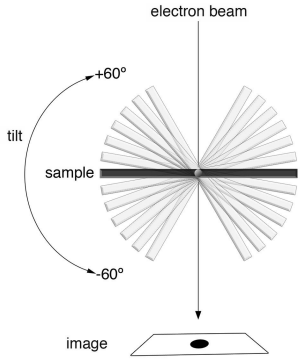
Abundance of undiscovered information

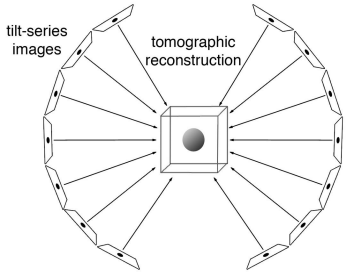
Imaging

Samples are imaged as they are tilted

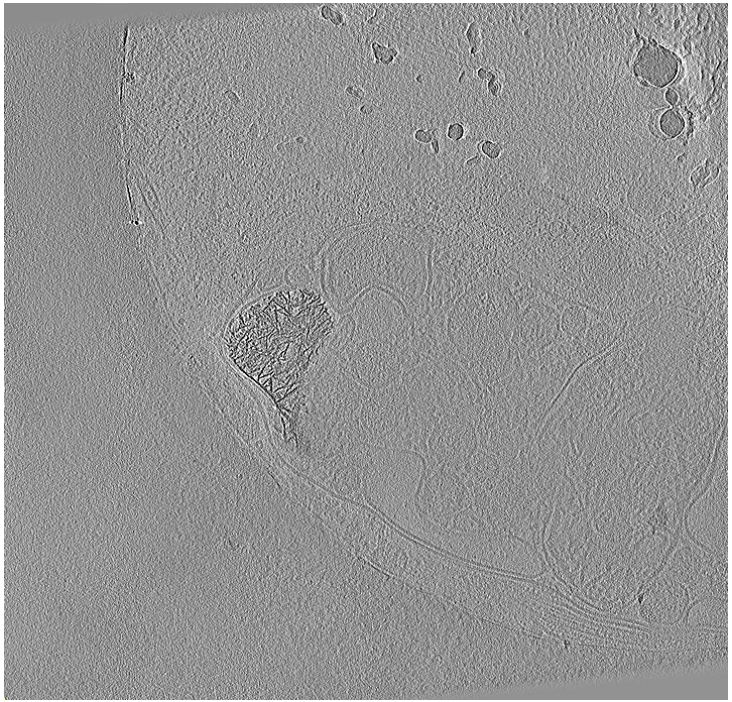
Angle restricted by thickness

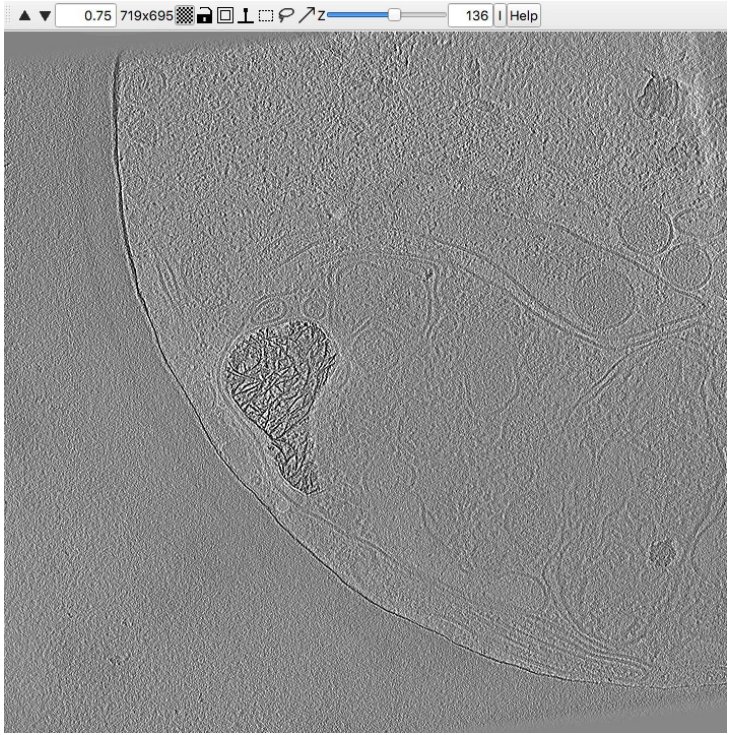
Series of 2D images are combined to produce a 3D reconstruction





Volumes





CryoEM process

Imaging

Alignment

Reconstruction

Denoising

Particle identification

Model generation

Problem

For Cryo-EM

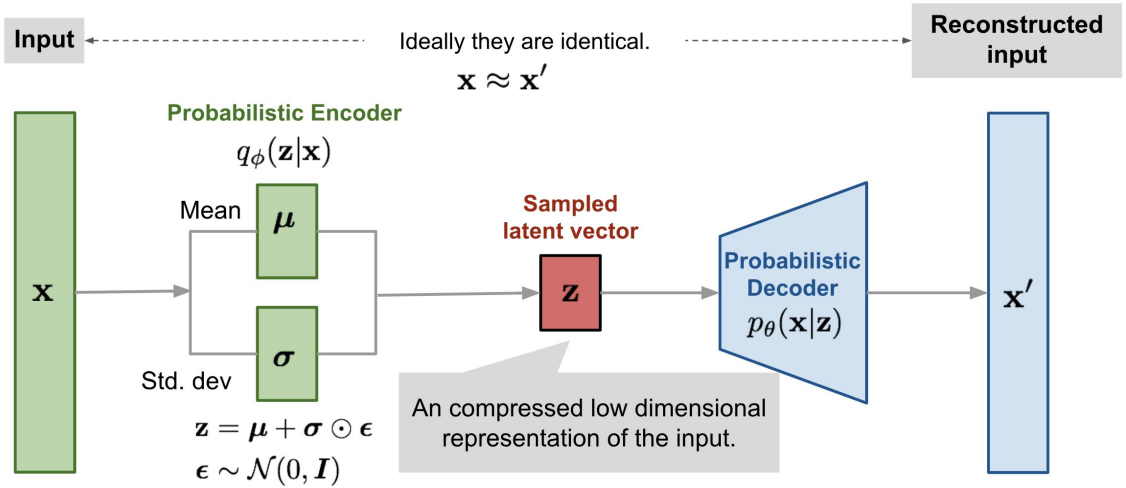
- Identify molecular components within the crowded cellular environment
- Large variation within each protein structure
- Identified particles can be used to generate protein structure models

Can we 3d segment features of interests at different levels of hierarchy with no supervision?

Initial approach

VAE based probabilistic approach

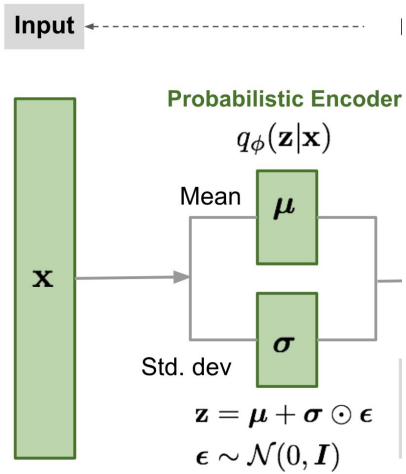
Learn semantically relevant representation of sampled 3D volumes



VAE

Encoder retrieves feature representation

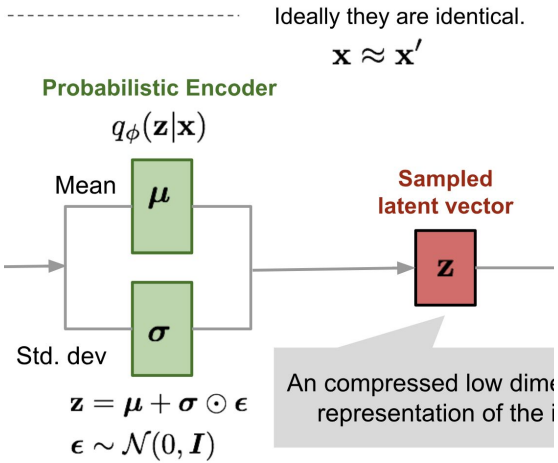
Downsamples input



VAE

Sample latent vector z

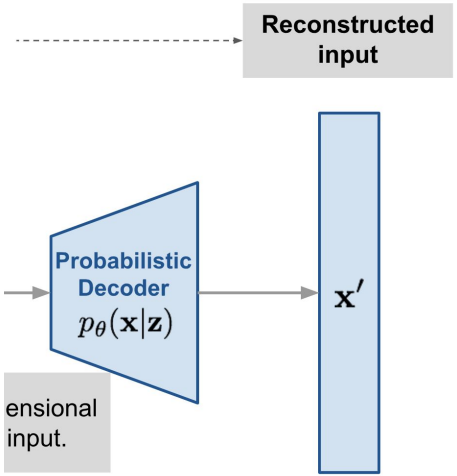
Learn to approximate true p(z|x) with encoder q(z|x)



VAE

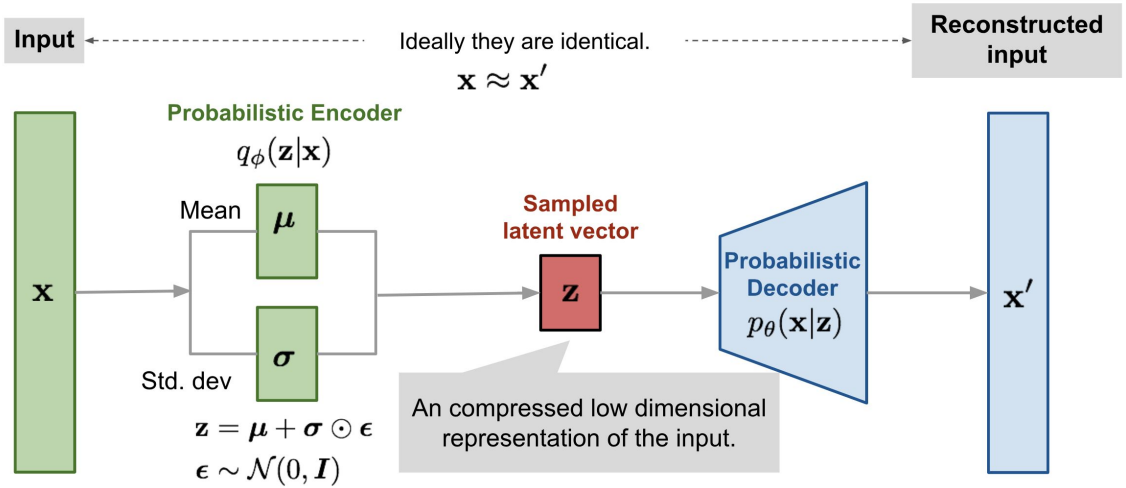
Decoder reconstructs input from latent vector

Reconstruction ensures semantic features kept



Initial approach

Learn downsampled representation of input



Probabilistic VAE pros

Can learn robust and generalizable representations

Can qualitatively visualize representations

Can create potential segmentations

Segmentation results

Use qualitative visualizations to measure performance

Utilize plotly to showcase sampled 3D visualizations after clustering

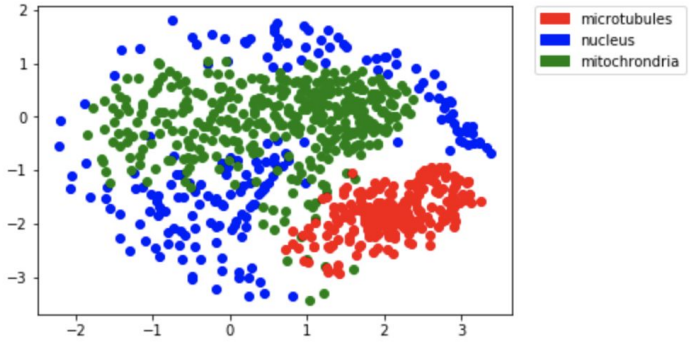
Representation space visualizations

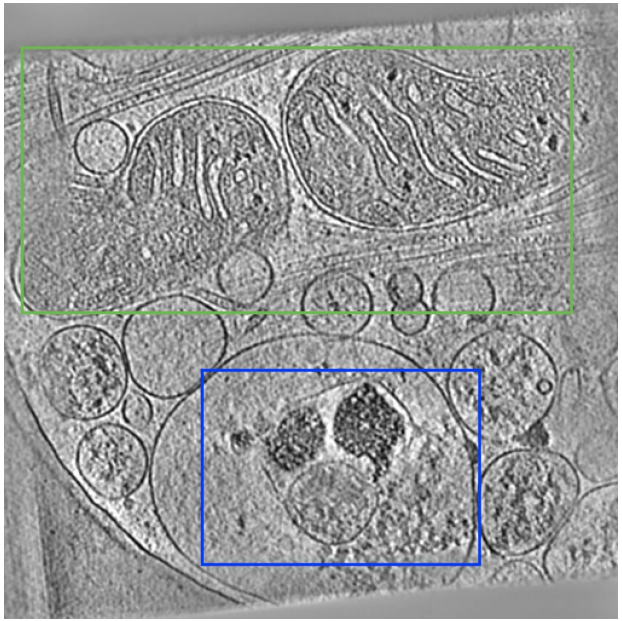
Representation μ generated of dimension 64

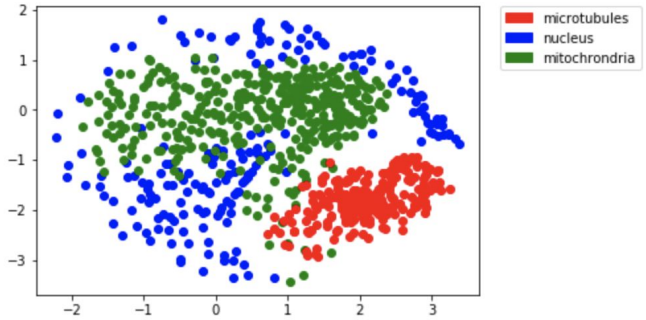
Apply dimension reduction through PCA & uniform manifold approximation

Representation of nucleus & mitochondria & microtubules

Out[31]: [<matplotlib.lines.Line2D at 0x7fc6b7b09690>]





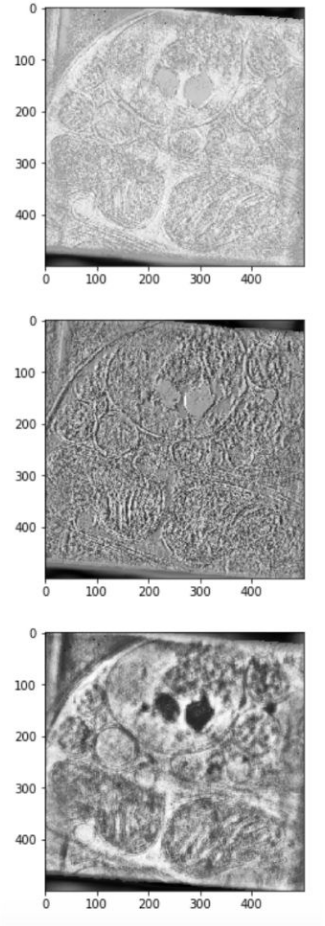

microtubules
nucleus
mitochrondria
2
1
0
−1
−2
−3
−2
−1
0
1
2
3

Component analysis

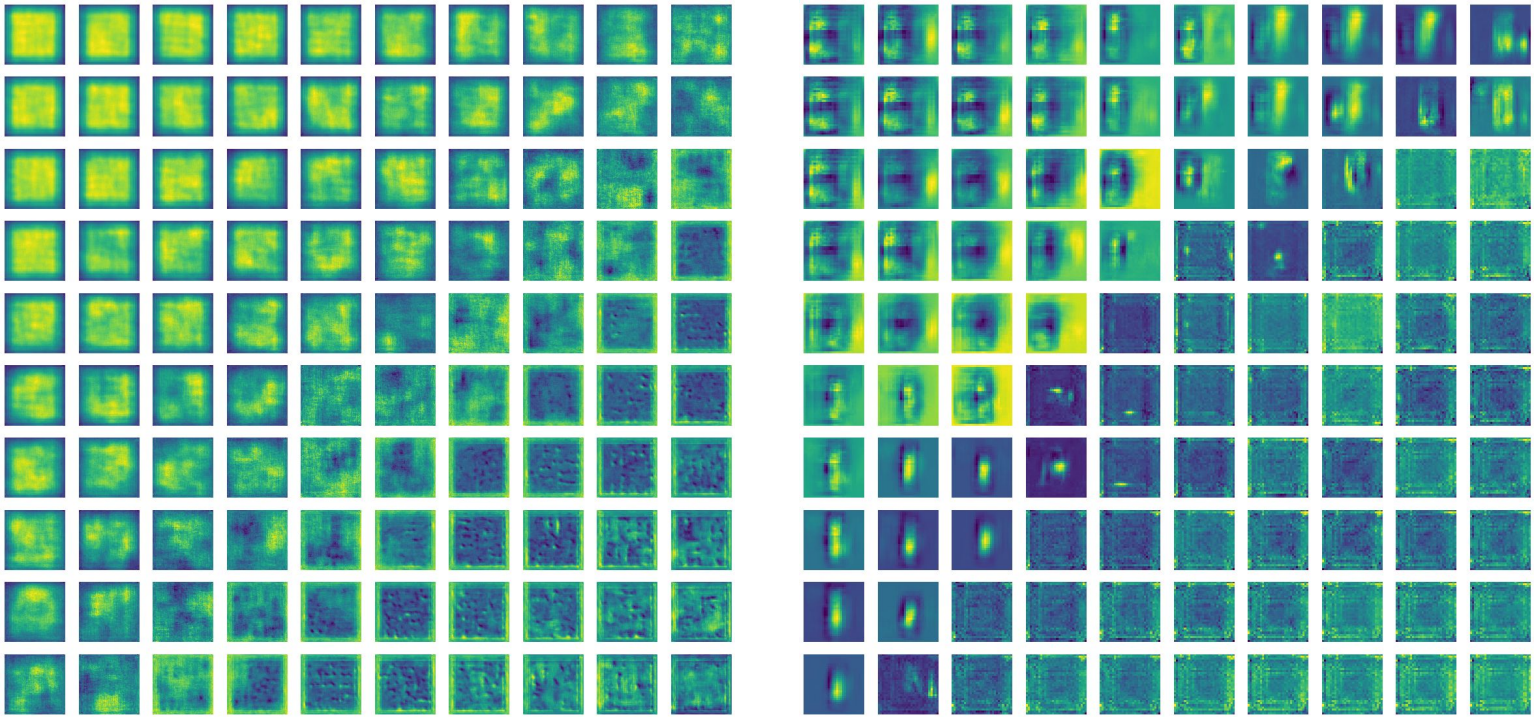
Representation μ of dimension 3

Sliding window inference

Component should represent higher level feature information



Latent space reconstructions



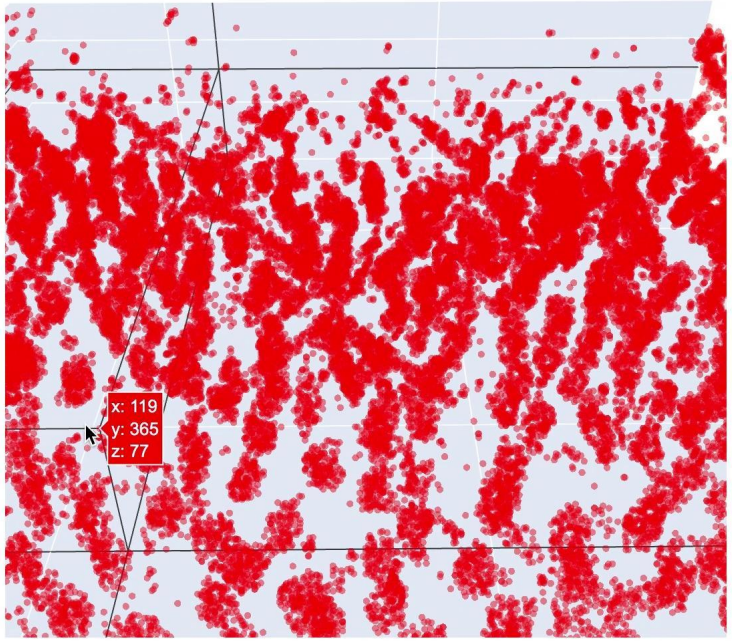
Apoferritin

Clustering on representation space

High sensitivity for apoferritin

Noise around detected instances

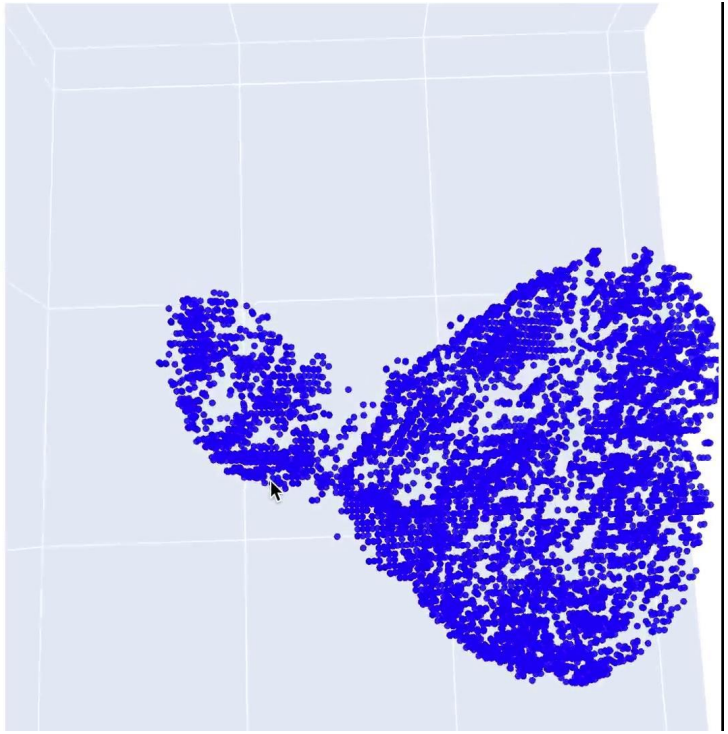
Incorrectly identified gold fiducials ‘X’s



Neuronal cells

Identified protein aggregate

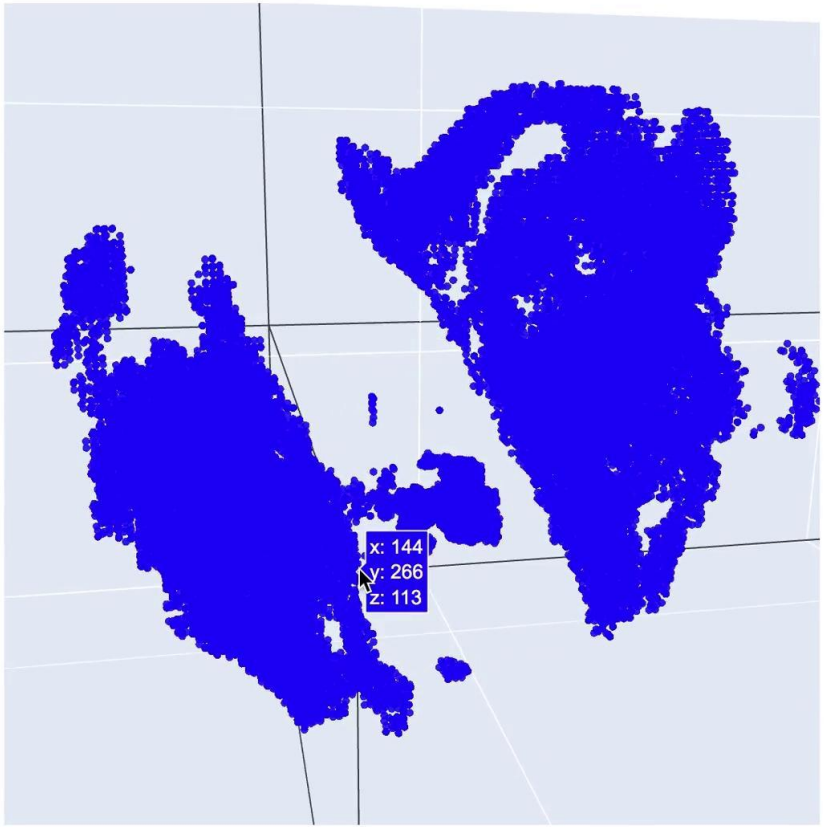
Can visualize structure



Neuronal cells

Learning interesting structure

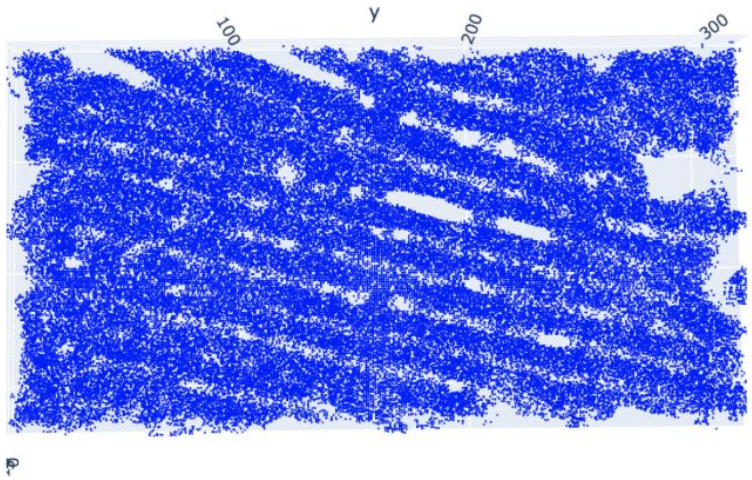
Difficult to capture semantic information



Microtubules

Noise within segmentation

Requires post-processing



Process

Representation space

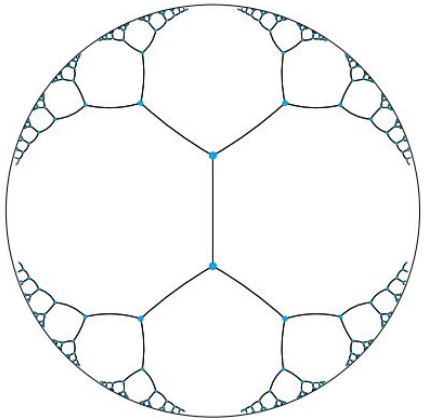
Component understanding

Segmentation for objects of interest

Need to learn more natural representation

Representation learning

How do we learn the most effective and natural form of representation μ?



Hyperbolic representation learning

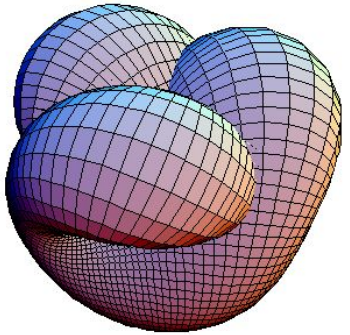
Preliminaries

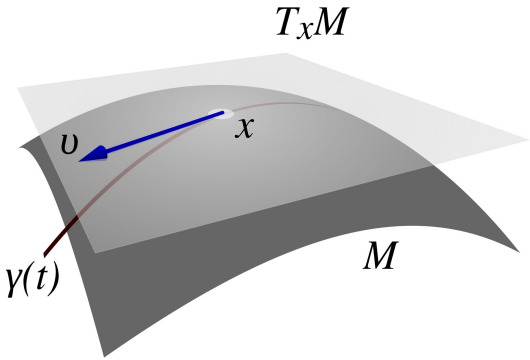
Introduction to manifolds

A **manifold** M of dimension d is an object where the neighborhood of every point locally resembles d-dimensional Euclidean space (R^d)

Define a **tangent space** $T_z M$ for each point z on M

Think of a manifold as the generalization of a 2D surface to n dimensions

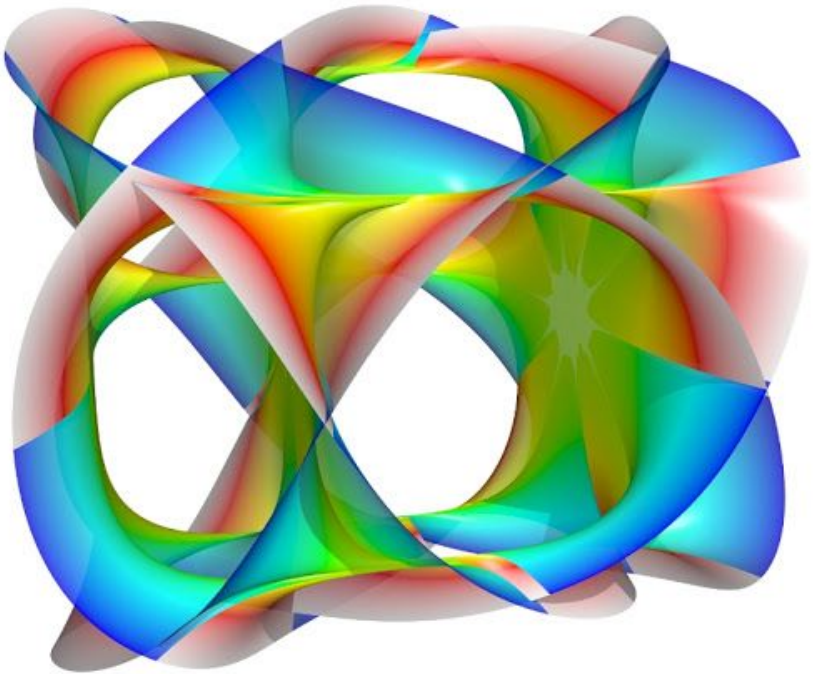




Riemannian manifolds

A **Riemannian manifold** is a manifold M with a metric g at every point x

The metric g allows us to locally define angles, distance, & area on M



Riemannian manifolds

Define the inner product at point z

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

Define a norm

$$\mathcal{T}_z\mathcal{M}: \|\cdot\|_z = \sqrt{\langle \cdot, \cdot \rangle_z}$$

Riemannian manifolds

From the matrix representation G define areas

$$d\mathcal{M}(z) = \sqrt{|G(z)|}dz$$

The length of a curve $\gamma : t \mapsto \gamma(t) \in \mathcal{M}$ defined as

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

Define the global distance on M as

$$d(x, y) = \inf_{\gamma} \int_0^1 \sqrt{g_{\gamma(t)}(\dot{\gamma}(t), \dot{\gamma}(t))}dt$$

The shortest path connecting two points on a manifold is called a **geodesic**

Riemannian manifolds

Define the **exponential map** as

$$\exp_{\boldsymbol{z}}(\boldsymbol{v}) = \gamma(1)$$

Riemannian Manifolds

Define the **logarithm map** as the inverse of the exponential map

$$\log_{\boldsymbol{z}} = \exp_{\boldsymbol{z}}^{-1} : \mathcal{M} \to \mathcal{T}_{\boldsymbol{z}}\mathcal{M}$$

Warning: the logarithm map doesn't always exist!

Poincare ball model

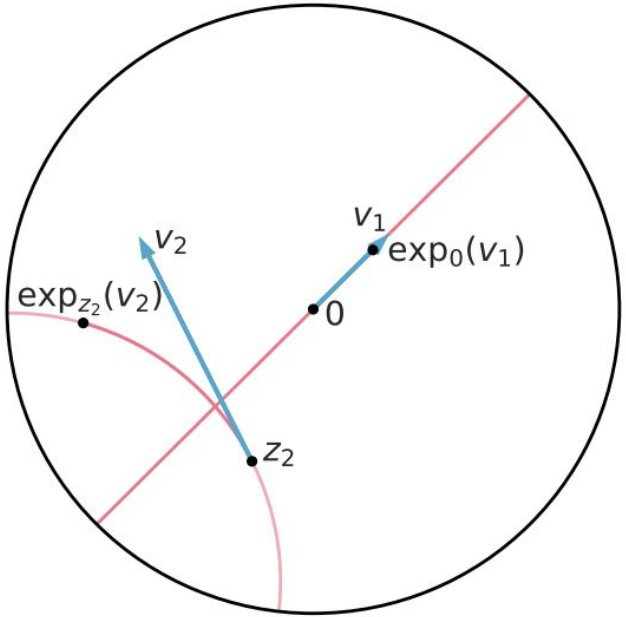
The Poincare ball model is one of five equivalent models of constant curvature *hyperbolic space*

The Poincare ball has closed form equations for all quantities of interest

Formally the Poincare ball is defined as $\mathbb{B}_c^d = (\mathcal{B}_c^d, \mathfrak{g}_p^c)$.

The second term is the **metric tensor**

$$\mathfrak{g}_p^c(z) = (\lambda_z^c)^2 \, \mathfrak{g}_e(z),$$
$$\lambda_z^c = \frac{2}{1 - c\|z\|^2}$$



Gyrovector operations

Mobius addition

$$\boldsymbol{z} \oplus_c \boldsymbol{y} = \frac{(1 + 2c\,\langle \boldsymbol{z}, \boldsymbol{y} \rangle + c\|\boldsymbol{y}\|^2)\boldsymbol{z} + (1 - c\|\boldsymbol{z}\|^2)\boldsymbol{y}}{1 + 2c\,\langle \boldsymbol{z}, \boldsymbol{y} \rangle + c^2\|\boldsymbol{z}\|^2\|\boldsymbol{y}\|^2}.$$

Exponential map

$$\exp_{\boldsymbol{z}}^c(\boldsymbol{v}) = \boldsymbol{z} \oplus_c \left(\tanh\left(\sqrt{c} \frac{\lambda_{\boldsymbol{z}}^c \|\boldsymbol{v}\|}{2} \right) \frac{\boldsymbol{v}}{\sqrt{c}\|\boldsymbol{v}\|} \right)$$

Logarithm map

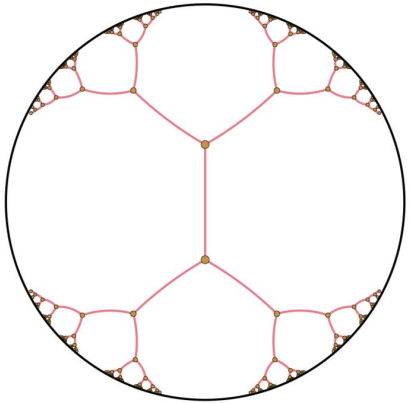
$$\log_{\boldsymbol{z}}^c(\boldsymbol{y}) = \frac{2}{\sqrt{c}\lambda_{\boldsymbol{z}}^c} \tanh^{-1}\left(\sqrt{c}\| -\boldsymbol{z} \oplus_c \boldsymbol{y}\| \right) \frac{-\boldsymbol{z} \oplus_c \boldsymbol{y}}{\| -\boldsymbol{z} \oplus_c \boldsymbol{y}\|}.$$

Method

Representation learning

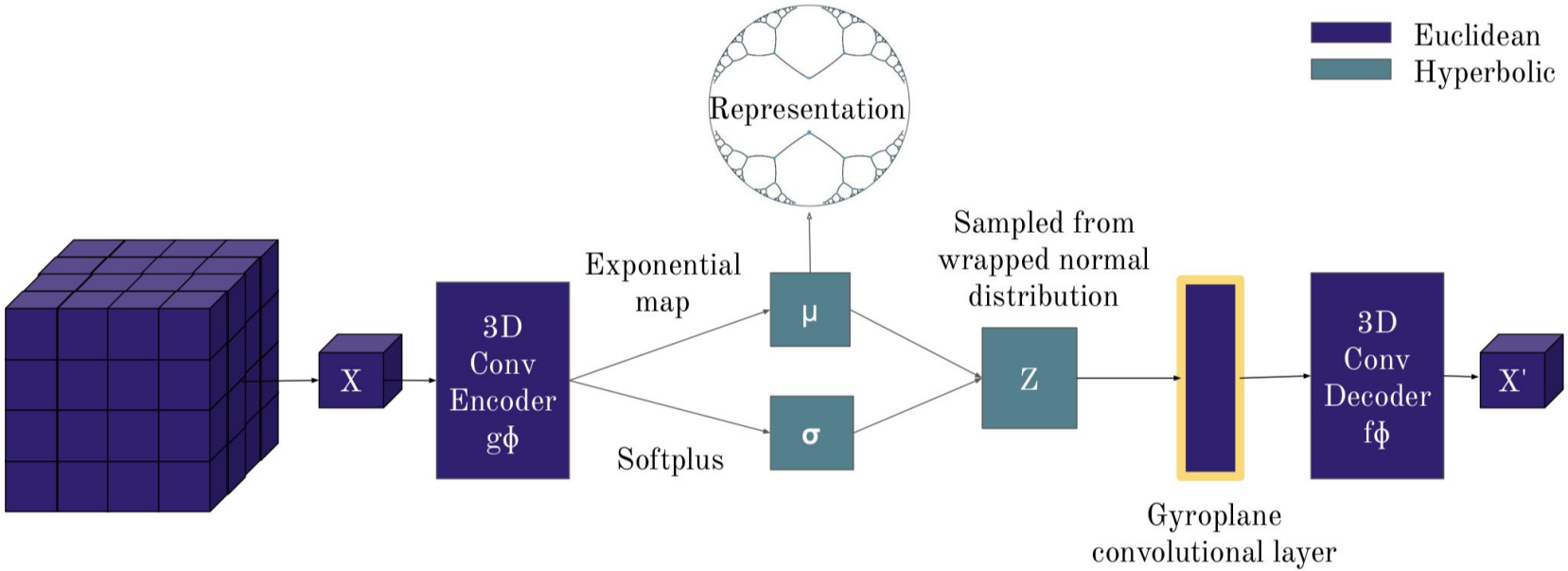
Hyperbolic representations effectively learn hierarchical structure

Can embed tree-like structure in hyperbolic spaces with arbitrarily low error



Hyperbolic VAE

Unsupervised representation learning

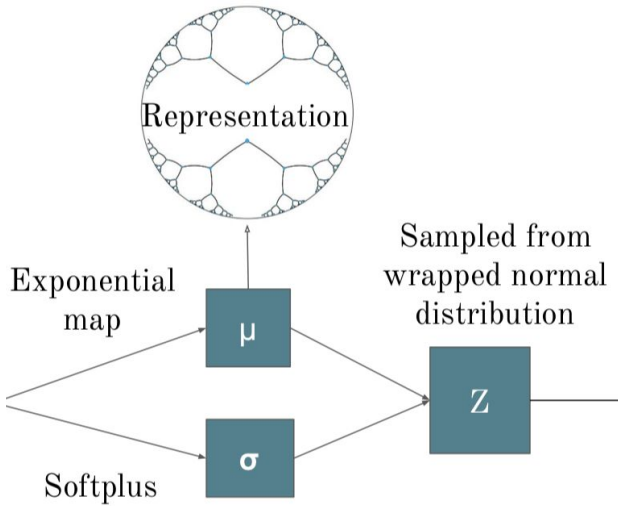


Hyperbolic VAE

μ = mean = ***representation***

σ = standard deviation

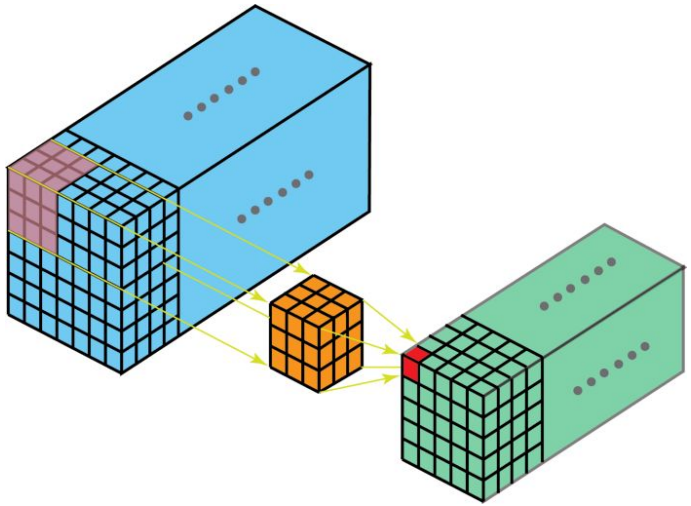
z = latent vector



Methods

Euclidean encoder with 3D convolutional layers to learn features

Dense layers output $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$



Methods

Map $\boldsymbol{\mu}$ into hyperbolic space through the **exponential map**

Apply softplus activation to $\boldsymbol{\sigma}$

$$f(x) = \ln(1 + e^x)$$

Sampling

$\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ parameterize the hyperbolic variational posterior

Need to sample with hyperbolic $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ to get latent vector z

Wrapped normal and Riemannian normal as prior and posterior distribution

Algorithm 1 Hyperbolic normal sampling scheme

Require: $\boldsymbol{\mu}$, σ^2, dimension d, curvature c
  if Wrapped normal **then** $\boldsymbol{v} \sim \mathcal{N}(\mathbf{0}_d, \sigma^2)$
  else if Riemannian normal **then**
    Let g be a piecewise exponential proposal
    while sample r not accepted **do**
      Propose $r \sim g(\cdot)$, $u \sim \mathcal{U}([0, 1])$
      if $u < \frac{\rho^{\mathrm{R}}(r)}{g(r)}$ **then** Accept sample r
    Sample direction $\boldsymbol{\alpha} \sim \mathcal{U}(\mathbb{S}^{d-1})$
    $\boldsymbol{v} \leftarrow r\boldsymbol{\alpha}$
  Return $\boldsymbol{z} = \exp^c_{\boldsymbol{\mu}}\left(\boldsymbol{v}/\lambda^c_{\boldsymbol{\mu}}\right)$

Riemannian normal

Distribution maximizing entropy given expectation and variance

$$\mathcal{N}^{\mathrm{R}}_{\mathbb{B}^d_c}(\boldsymbol{z}|\boldsymbol{\mu},\sigma^2) = \frac{d\nu^{\mathrm{R}}(\boldsymbol{z}|\boldsymbol{\mu},\sigma^2)}{d\mathcal{M}(\boldsymbol{z})} = \frac{1}{Z^{\mathrm{R}}}\exp\left(-\frac{d^c_p(\boldsymbol{\mu},\boldsymbol{z})^2}{2\sigma^2}\right)$$

Wrapped normal

Push forward measure mapping normal distribution along the exponential map

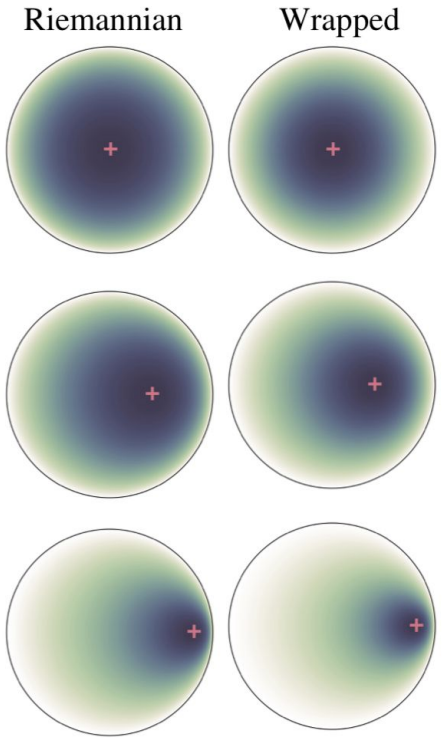
$$\mathcal{N}^{\mathrm{W}}_{\mathbb{B}^d_c}(\boldsymbol{z}|\boldsymbol{\mu},\Sigma) = \frac{d\nu^{\mathrm{W}}(\boldsymbol{z}|\boldsymbol{\mu},\Sigma)}{d\mathcal{M}(\boldsymbol{z})}$$

$$= \mathcal{N}\left(\lambda^c_{\boldsymbol{\mu}} \log_{\boldsymbol{\mu}}(\boldsymbol{z}) \middle| \mathbf{0}, \Sigma\right) \left(\frac{\sqrt{c}\, d^c_p(\boldsymbol{\mu},\boldsymbol{z})}{\sinh(\sqrt{c}\, d^c_p(\boldsymbol{\mu},\boldsymbol{z}))}\right)^{d-1}$$

Distributions on the Poincare ball

Both normal distributions are similar

Riemannian normal has slightly larger mode



Methods

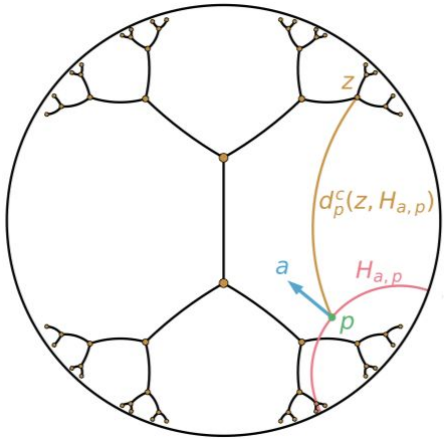
Decoder uses **gyroplane layer** to map hyperbolic representation to Euclidean

On the Poincare ball a gyroplane layer is defined as

$$f^c_{\boldsymbol{a},\boldsymbol{p}}(\boldsymbol{z}) = \operatorname{sign}(\langle \boldsymbol{a}, \log^c_{\boldsymbol{p}}(\boldsymbol{z}) \rangle_{\boldsymbol{p}}) \, \|\boldsymbol{a}\|_{\boldsymbol{p}} \, d^c_p(\boldsymbol{z}, H^c_{\boldsymbol{a},\boldsymbol{p}})$$

With closed form distance equation as

$$d^c_p(\boldsymbol{z}, H^c_{\boldsymbol{a},\boldsymbol{p}}) = \frac{1}{\sqrt{c}} \sinh^{-1}\left(\frac{2\sqrt{c}|\langle -\boldsymbol{p} \oplus_c \boldsymbol{z}, a \rangle|}{(1 - c\| -\boldsymbol{p} \oplus_c \boldsymbol{z}\|^2)\|\boldsymbol{a}\|} \right)$$



Gyroplane convolutional layer

Encoder output $\boldsymbol{\mu}$ is produced by 3D convolutional layers

Hence both $\boldsymbol{\mu}$ and sampled vector z can be thought of a 3D volume

Each voxel is represented by a vector in hyperbolic space

Gyroplane convolutional layer

Define gyroplane 3D convolutional layer as the Euclidean 3D convolutional layer

Euclidean linear operation replaced by a gyroplane operation

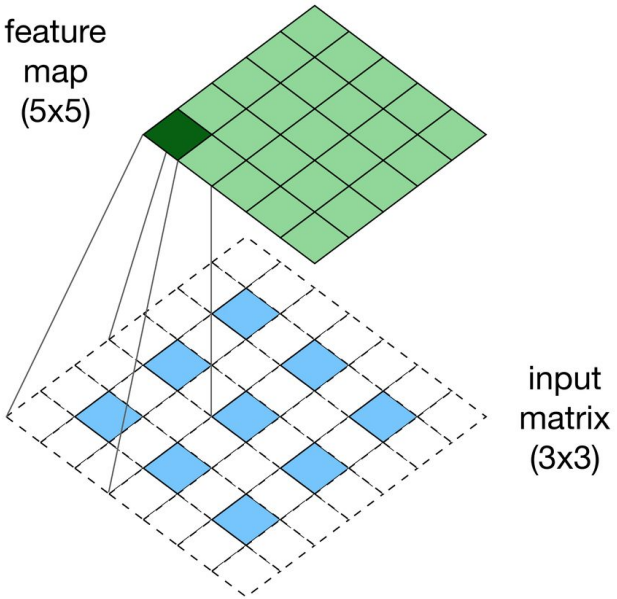
Gyroplane layer is a hyperbolic affine transformation from Poincare ball to Euclidean

$$y_{r,s,t} = \sum_{\alpha=r-\lfloor k/2 \rfloor}^{r+\lfloor k/2 \rfloor} \sum_{\beta=s-\lfloor k/2 \rfloor}^{s+\lfloor k/2 \rfloor} \sum_{\gamma=t-\lfloor k/2 \rfloor}^{t+\lfloor k/2 \rfloor} g_{w,b}(x_{\alpha,\beta,\gamma})$$

$$g_{a,p}(z) = \sum_k \lambda_p^2 \cdot \text{sign}((a \star \log_p(z[k]))) \cdot ||a|| \cdot d_p(z[k], H_{a,p})$$

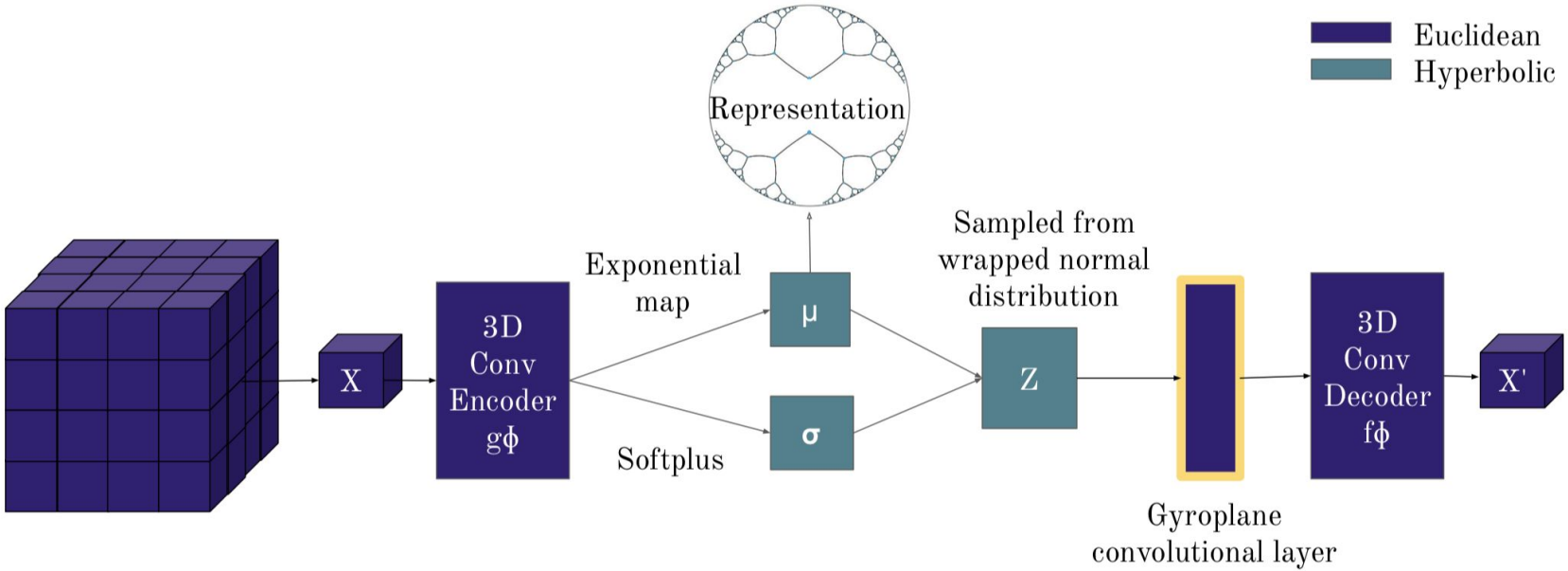
Methods

Decoder includes 3D up-convolutional layers back to input size



Hyperbolic VAE

Unsupervised representation learning



VAE Euclidean Objective

Goal is to learn true posterior of latent vector z given input x

Minimize KL divergence (information loss) between approximate posterior from encoder q(z|x) and true posterior p(z|x)

Maximizing ELBO (evidence lower bound) minimizes KL divergence without p(x)

$$ELBO(\lambda) = \mathbf{E}_q[\log p(x, z)] - \mathbf{E}_q[\log q_\lambda(z \mid x)]$$

$$ELBO_i(\theta, \phi) = \mathbb{E}q_\theta(z \mid x_i)[\log p_\phi(x_i \mid z)] - \mathbb{KL}(q_\theta(z \mid x_i) \mid\mid p(z))$$

Hyperbolic ELBO Objective

ELBO

$$\log p(\boldsymbol{x}) \geq \mathcal{L}_{\mathcal{M}}(\boldsymbol{x}; \theta, \phi) \triangleq \int_{\mathcal{M}} \ln \left(\frac{p_\theta(\boldsymbol{x}|\boldsymbol{z}) p(\boldsymbol{z})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})} \right) q_\phi(\boldsymbol{z}|\boldsymbol{x}) \, d\mathcal{M}(\boldsymbol{z})$$

Reparameterization through hyperbolic polar change of coordinates

$$\boldsymbol{z} = \exp_{\boldsymbol{\mu}}^{c} \left(G(\boldsymbol{\mu})^{-\frac{1}{2}} \, \boldsymbol{v} \right) = \exp_{\boldsymbol{\mu}}^{c} \left(\frac{r}{\lambda_{\boldsymbol{\mu}}^{c}} \boldsymbol{\alpha} \right)$$

Details in Appendix of *Continuous Hierarchical Representations with Poincaré Variational Auto-Encoders by Mathieu et al.*

Hierarchical objective

Enforce relations in different levels of hierarchy

Hierarchical triplet loss samples anchor parent, positive child, and negative child motivated by the compositional hierarchy of 3D volumes

Hierarchical objective

Encourages the anchor patch (parent) and a sub-patch (positive child) to have similar representations

In hyperbolic space this has the interpretation of belonging to the same hierarchy

Anchor patch and a distant patch (negative child) to have dissimilar representation

Belonging to the different hierarchies

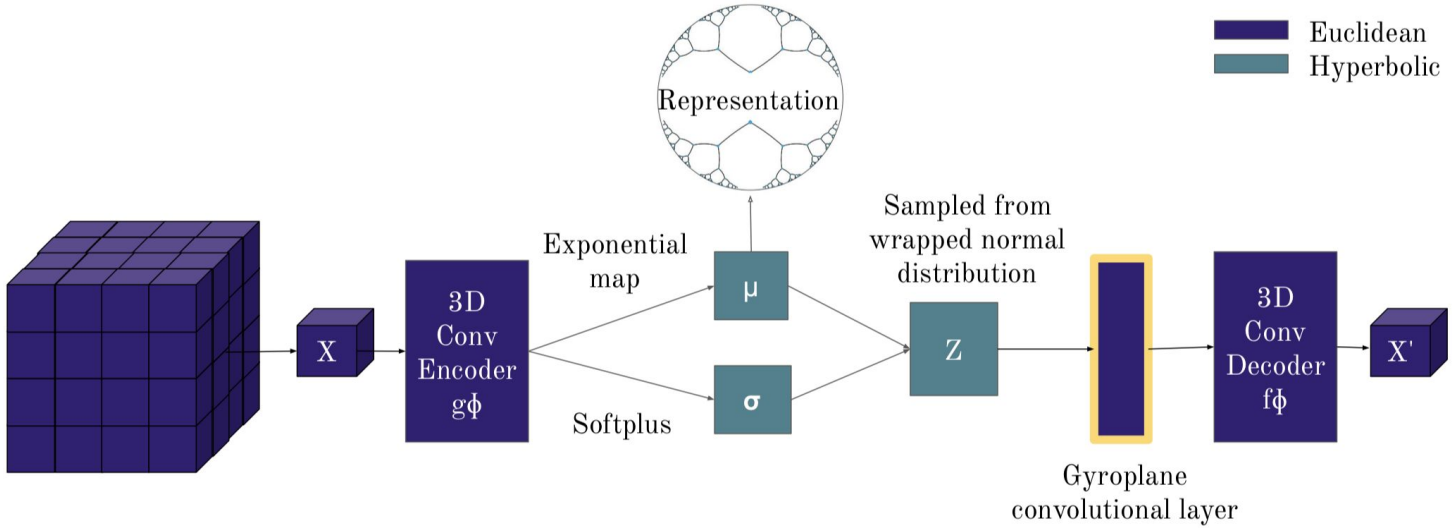
Hierarchical triplet loss

$$L_{\text{triplet}}(\mu_{\text{p}}, \mu_{\text{pos}}, \mu_{\text{neg}}) := \max(0, d_p(\mu_{\text{p}}, \mu_{\text{pos}}) - d_p(\mu_{\text{p}}, \mu_{\text{neg}}) + \alpha)$$

$$d_p(x, y) = \cosh^{-1}\left(1 + 2\frac{||x - y||^2}{(1 - ||x||^2)(1 - ||y||^2)}\right)$$

Loss

$$L_{\text{total}} = L_{\text{ELBO}} + \beta L_{\text{triplet}}$$



Experiments

Clustering

Retrieve representation of sampled patches from input volume through a sliding window

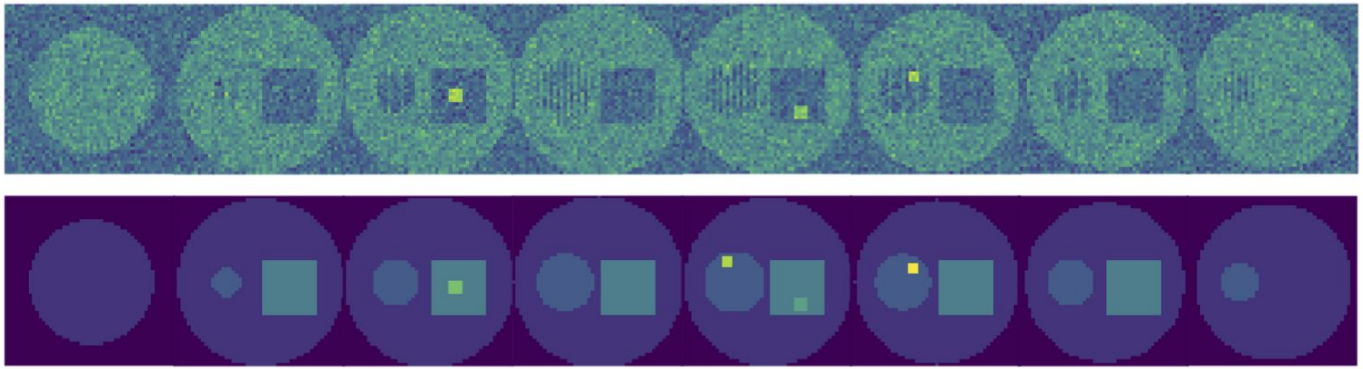
Perform hyperbolic kmeans

Use Hungarian algorithm for assignment

Evaluation

$$Dice = \frac{2 \times TP}{(TP + FP) + (TP + FN)}$$

Biologically-inspired toy dataset



Prior approaches

	Dice @ *Level 1*	Dice @ *Level 2*	Dice @ *Level 3*	Supervision type
Cicek et al [9]	0.96843	0.82867	0.66771	3D Semi-supervised
Zhao et al [40]	0.98892	0.65478	0.35651	3D Semi-supervised
Nalepa et al [28]	0.53017	0.27612	0.11997	3D Unsupervised
Ji et al [16]	0.58866	0.29086	0.14999	2D to 3D Unsupervised
Moriya et al [26]	0.62777	0.31120	0.14131	3D Unsupervised
Ours	**0.95211**	**0.54065**	**0.21623**	3D Unsupervised

Ablations

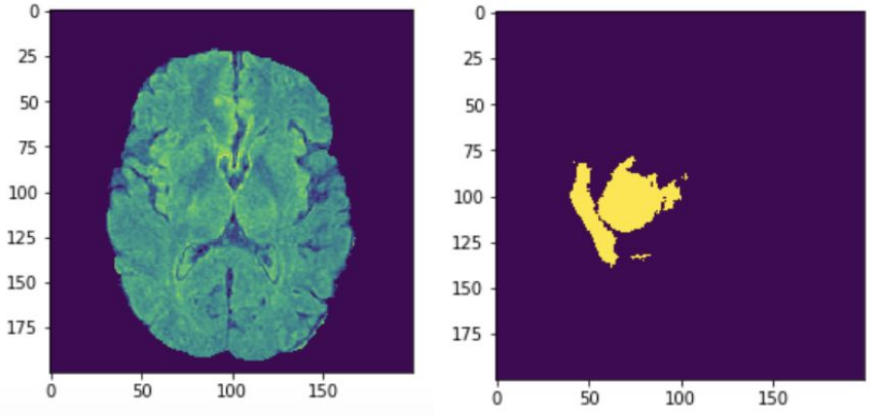
Latent Space	Configuration	Dice @ *Level 1*	Dice @ *Level 2*	Dice @ *Level 3*
Euclidean	Base	0.78370	0.32170	0.10890
	Triplet	0.76111	0.34202	0.15349
Hyperbolic	Base	0.83231	0.35185	0.13528
	GyroConv	0.90468	0.47297	0.20363
	Triplet	0.94522	0.53392	**0.22217**
	GyroConv & Triplet	**0.95211**	**0.54065**	0.21623

BRATS dataset

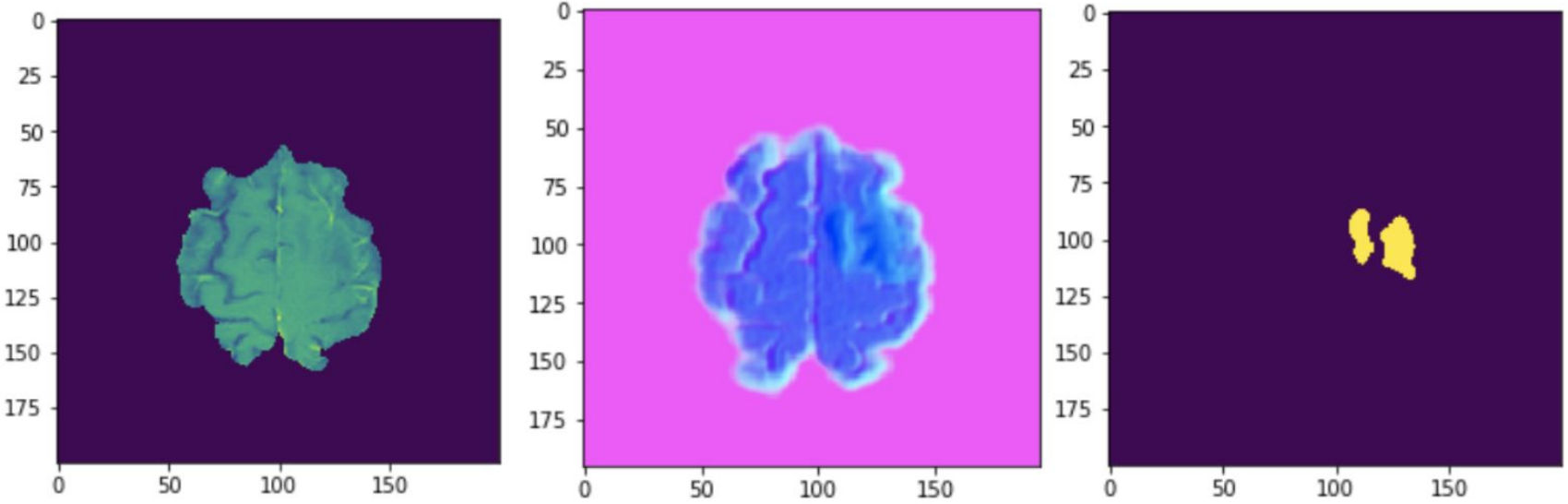
3D MRI scans of the brain

Annotated with tumors

Output tumor masks on volume



BRATS results

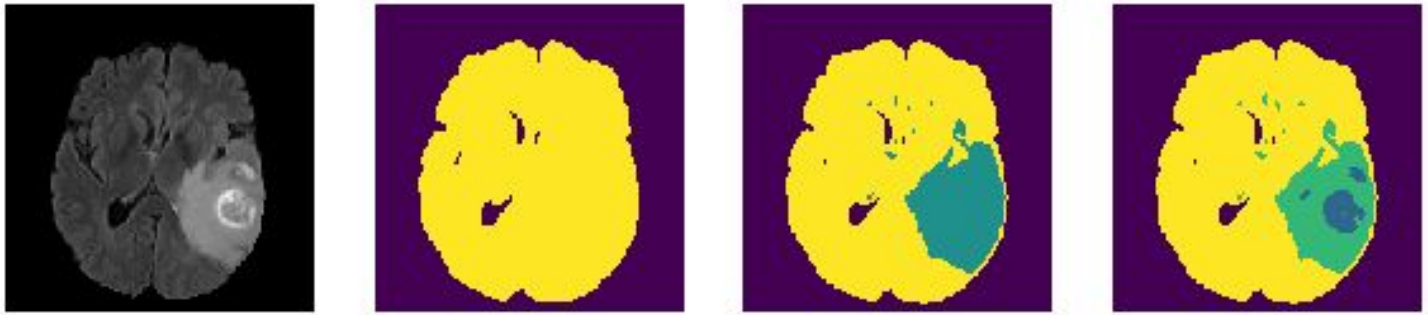


Prior approaches

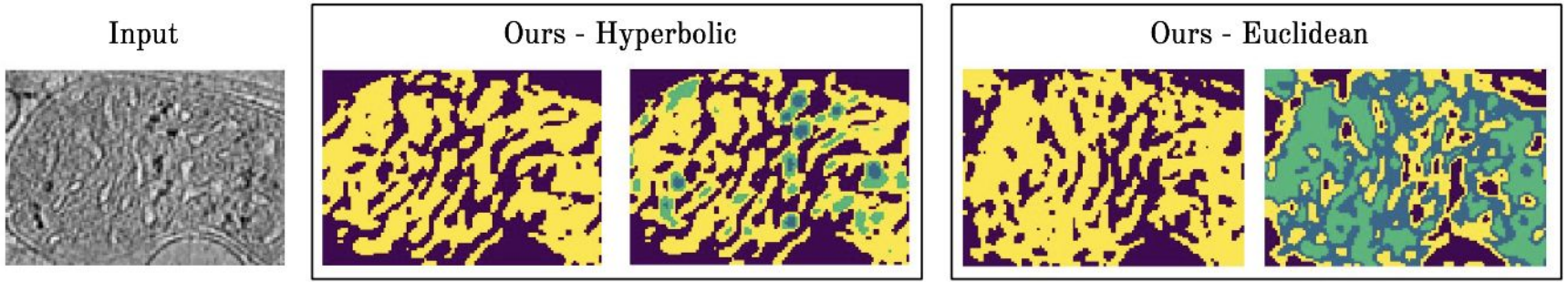
BraTS dataset	Dice WT	Algorithm type
SOTA 4 modalities [17]	0.88796	3D Fully-supervised
Zhao et al [40]	0.64826	3D Semi-supervised
Cicek et al [9]	0.75965	3D Semi-supervised
Ji et al [16]	0.21076	2D to 3D Unsupervised
Moriya et al [26]	0.42515	3D Unsupervised
Nalepa et al [28]	0.49503	3D Unsupervised
Ours	**0.68391**	3D Unsupervised

Hierarchical segmentations

Can retrieve different levels of segmentations given one model



CryoEM



Joy Hsu joycj@stanford.edu
Jeff Gu jeffgu@stanford.edu
Michael Zhang mzhang20@stanford.edu
Serena Yeung syyeung@stanford.edu

Thanks to Gong Her Wu, Jesus Montoya, David Chmielewski, Michael Schmid, Wah Chiu from SLAC

Questions?