

Seg-HGNN: Unsupervised and Light-Weight Image Segmentation with Hyperbolic Graph Neural Networks

Debjyoti Mondal
d.mondal@samsung.com

Rahul Mishra
mishra.rl@samsung.com

Chandan Pandey
chandan.p@samsung.com

Samsung R&D Institute India
Bangalore

1 Training Environment

All our experiments were performed on an AMD Ryzen5 5600H 6-core CPU, with 8GB of RAM and an NVIDIA GTX 1650 with 4GB of VRAM. Table 1 lists the versions of all required packages.

2 Hyperparameters

In Table 2, we have provided the values of all hyperparameters. We use $k = 2$ for background - foreground segmentation, and $k = 4$ for semantic art segmentation on the foreground. Both these configurations use the same parameters except the number of epochs the model is trained for, which is 10 and 100 respectively.

Patch-size p refers to the p^2 pixels that form a patch. And stride s is the number of pixels to move to get to the next patch. Throughout all our experiments, we ensure that $p = s$. This ensures that none of the patches overlap, and also the number of nodes remain reasonable.

3 Visualizing Hyperbolic embeddings

On the Poincaré ball model, distance from the origin gives a natural idea of the classification confidence of patches. We use this property to judge the confidence of our clustering objective [1].

4 Samples

In the following pages, we attach a few samples for the reader to get an idea of Seg-HGNN's performance quality.

Package	Version
torch	1.13.0+cu116
torchvision	0.14.0+cu116
Pillow	9.2.0
opencv-python	4.8.1.78
numpy	1.23.2
matplotlib	3.8.1
scikit-learn	1.3.2

Table 1: Important packages and respective versions used.

Algorithm 1 Seg-HGNN**Input:** Image I , resolution (m, n) , patch size p , stride s , number of clusters k .**Output:** Assigned clusters to each segment \mathcal{S} .

- 1: $I \leftarrow \text{resize}(I, m, n)$
- 2: $f \leftarrow \text{ViT}(I, p, s) \in \mathbb{R}^{\frac{mn}{p^2} \times d}$ ▷ Extract features.
- 3: $w \leftarrow \frac{f f^T}{\frac{mn}{p^2}} \cdot (f f^T > 0) \in \mathbb{R}^{\frac{mn}{p^2} \times \frac{mn}{p^2}}$ ▷ Get adj.
- 4: $f^{\mathcal{L}} \leftarrow \exp(0 \| f)$ ▷ Project to Lorentz space.
- 5: **for** each epoch **do**
- 6: $\mathcal{F} \leftarrow \text{HyperbolicGCN}(f^{\mathcal{L}}, w)$
- 7: $\mathcal{F}' \leftarrow \log_{o_{\mathcal{C}}}(\mathcal{F})$ ▷ Bring back to Euclidean.
- 8: $\mathcal{F}'' \leftarrow \text{FullyConnected}(\mathcal{F}')$
- 9: $\mathcal{S} \leftarrow \text{softmax}(\mathcal{F}'')$
- 10: $\text{loss} \leftarrow \text{loss}_{\text{N-Cut}}(\mathcal{S}, w, k)$
- 11: Calculate gradients and update parameters.
- 12: **end for**

Hyperparameter	Value	
	$k = 2$	$k = 4$
epochs (for segmentation)	10	100
euclidean optimizer	Adam	
euclidean learning rate	0.01	
stiefel optimizer	Riemannian SGD	
stiefel learning rate	0.1	
dim	16	
resize resolution (m, n)	(280, 280)	
patch-size p	8	
stride s	8	

Table 2: Values of hyperparameters.

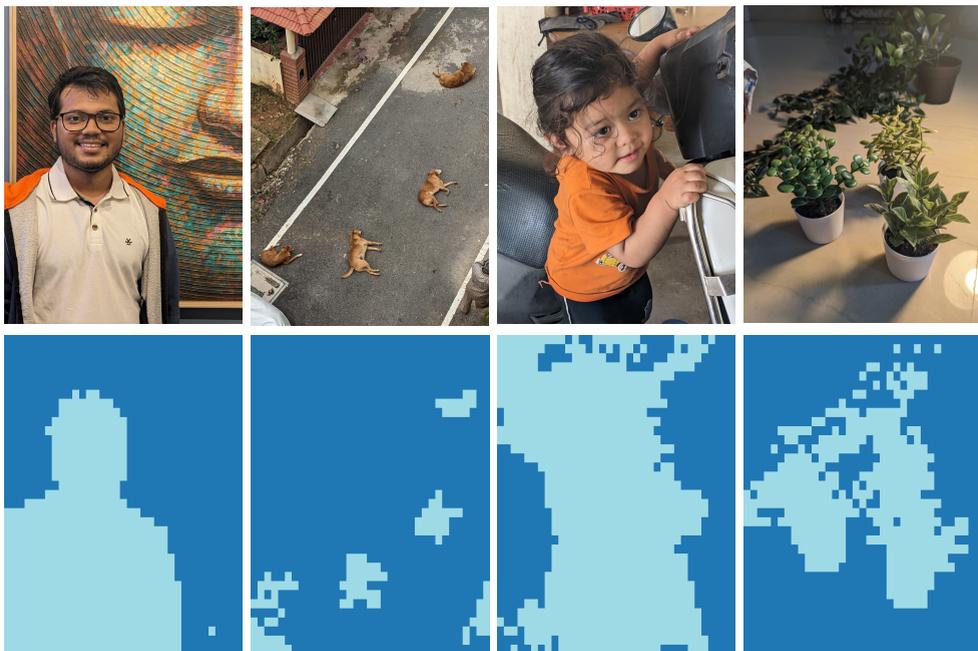


Figure 1: Object segmentation result using Seg-HGNN. The last two samples show some failure cases.

References

- [1] Maximillian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. *Advances in neural information processing systems*, 30, 2017.

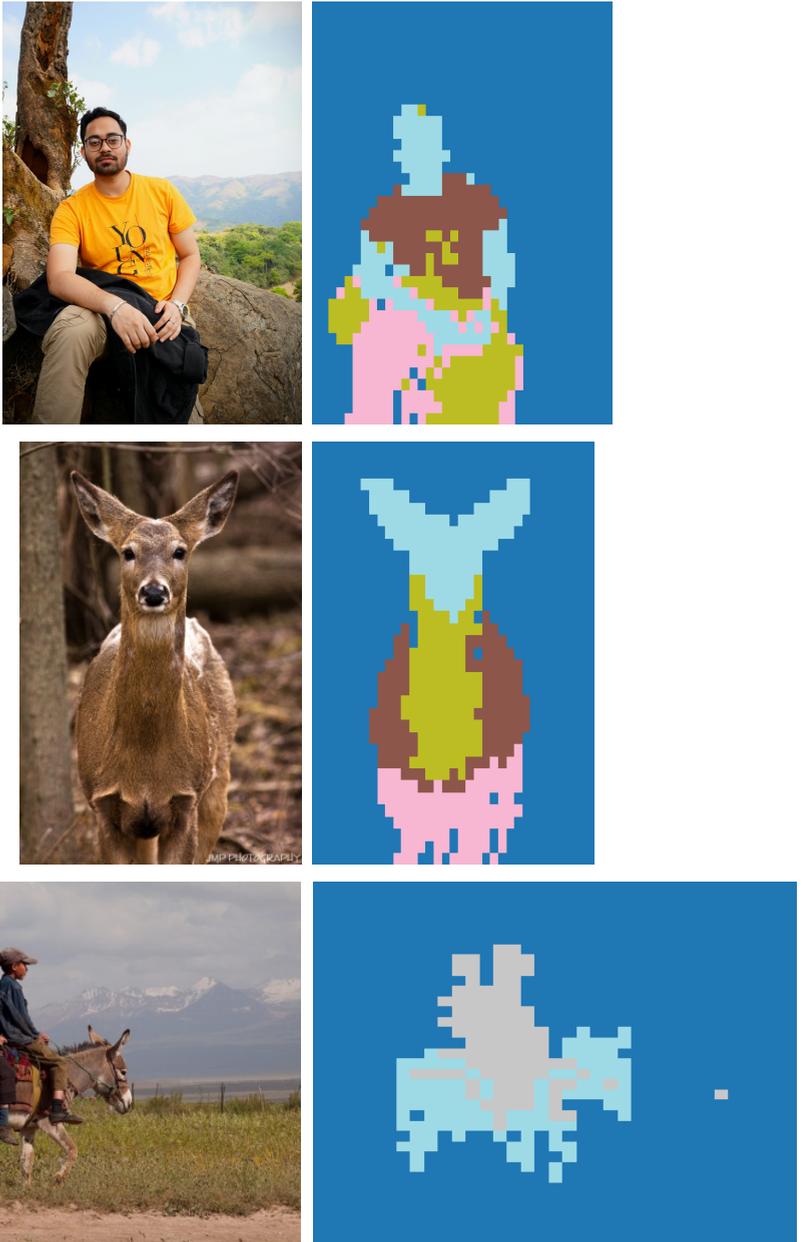


Figure 2: Semantic segmentation result using Seg-HGNN.