HyperPG: Probabilistic Prototypes on Hyperspheres for Interpretable Deep Learning

Maximilian Xiling Li[1] Korbinian Rudolf[1] Paul Mattes[1] Nils Blank[1] Rudolf Lioutikov[1,2]
[1]Intuitive Robots Lab, Karlsruhe Institute of Technology, Germany
[2]Robotics Institute Germany (RIG)
{m.li, lioutikov}@kit.edu

Figure 1. Similarity computation between prototype $\boldsymbol{p}$ and latent vector $\boldsymbol{z}$ for each formulation. Euclidean: L_2 distance. Hyperspherical: cosine similarity. Gaussian: Euclidean-space PDF. HyperPG: PDF over cosine similarities on the hypersphere surface.

Abstract

Prototypical part networks provide interpretable alternatives to black-box models by learning visual prototypes for classification. We present a systematic comparison of point-based and probabilistic prototype formulations across Euclidean and hyperspherical latent spaces, and introduce HyperPG: a Gaussian prototype representation on hyperspheres. Experiments on CUB-200-2011, Stanford Cars, and Oxford Flowers show that hyperspherical prototypes outperform Euclidean formulations while remaining robust to hyperparameter choices, reducing the tuning burden that plagues Euclidean approaches.

1. Introduction

Deep learning achieves strong performance in computer vision, yet the opacity of these models hinders deployment in safety-critical domains. Explainable AI (XAI) addresses this through post-hoc methods such as LIME [15], SHAP [6], and GradCAM [20], or through inherently interpretable architectures. Post-hoc explanations, however, may not reflect a model's true decision process [16].

Deep prototype networks such as ProtoPNet [1] and its variants [2, 17, 19] achieve inherent interpretability by learning a prototype layer whose predictions are based on feature-to-prototype distances. The choice of prototype formulation shapes latent structure: cosine similarity induces hyperspherical geometry with classification benefits [8], while probabilistic formulations enable Bayesian downstream tasks. Despite growing architectural diversity, no systematic comparison of prototype formulations exists.

This paper compares point-based and probabilistic prototype formulations across Euclidean and hyperspherical spaces, with the following contributions:

- **Prototype Formulation Overview:** A systematic comparison of learnable prototype definitions (Figure 1), covering point-based and probabilistic variants.
- **HyperPG:** A probabilistic prototype modeling Gaussians over cosine similarities with learned anchor α, mean μ, and variance σ^2, projected onto a hypersphere.
- **Benchmarking:** Classification experiments on CUB-200-2011 [22], Stanford Cars [4], and Oxford Flowers [12], including a robustness analysis under simplified training. The code is published via GitHub https://github.com/LiXiling/prob-proto.

2. Related Work

Prototype Learning. Prototype learning for image classification was pioneered by autoencoder-based approaches

Table 1. Overview of 15 related models and their prototype learning configurations.

Model	Similarity	Shape	Assignment	Clf. Head	
Prototype Decoder	Euclidean	Entire Image	Class Exclusive	FCL	[5]
ProtoPNet	Euclidean	Patch	Class Exclusive	FCL	[1]
Def. ProtoPNet	**Cosine**	Spatial Arrangement	Class Exclusive	FCL	[2]
ProtoPShare	Euclidean	Patch	Merged after Training	FCL	[17]
ProtoPool	Euclidean	Patch Pooling	Shared	FCL	[18]
TesNet	**Grassman**	Patch	Class Exclusive	FCL	[25]
ProtoTree	Euclidean	Patch	Shared	Decision Tree	[10]
ProtoKNN	**Cosine**	Patch	Class Exclusive	KNN Clf	[21]
PIPNet	**Cosine**	Patch	Shared	FCL	[11]
ST-ProtoPNet	**Cosine**	Patch	Class Exclusive	Branch Aggregation	[23]
LucidPPN	Euclidean	Separate Color & Texture	Class Exclusive	Branch Aggregation	[13]
ProtoSeg	**Cosine**	Patch	Class Exclusive	FCL	[27]
ProtoGMM	**Gaussian**	Patch	Class Exclusive	FCL	[9]
MGProto	**Gaussian**	Patch	Class Exclusive	Bayesian Likelihood	[24]
ProtoPFormer	Euclidean	Transformer Token	Class Exclusive	Branch Aggregration	[26]

[5], where prototypes represent entire images. ProtoPNet [1] introduced *prototypical parts* - class-specific latent patches compared to image features via Euclidean distance. Subsequent work extends this in various directions: ProtoPShare [17] and ProtoPool [18] share prototypes across classes; Deformable ProtoPNet [2] uses spatial mixtures with cosine similarity; TesNet [25] computes similarity on the Grassmann manifold. Alternative classifiers replace the linear head with decision trees [10], k-NN [21], or support-vector-inspired boundaries [23]. PIPNet [11] and LucidPPN [13] focus on learning more meaningful prototype features.

Probabilistic and Hyperspherical Prototypes. Viewing prototype learning as latent-space clustering [27] motivates probabilistic formulations; ProtoGMM [9] and MGProto [24] model prototypes as Gaussians in Euclidean space. ProtoPFormer [26] adapts ProtoPNets to ViT [3] backbones. Cosine similarity is known to benefit classification via hyperspherical structure [8], yet probabilistic prototypes in hyperspherical space remain unexplored. We address this gap with HyperPG, combining the strengths of both (Table 1).

3. Prototype Formulations

Prototype learning finds structure in latent representations by comparing inputs to retained prototype vectors. We overview existing formulations (illustrated in Figure 1) and introduce HyperPG.

Let $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}$ be a labeled training set with classes C. Each class c is represented by Q prototypes $\boldsymbol{P}_c = \{\boldsymbol{p}_{c,j}\}_{j=1}^{Q}$. An encoder Enc maps inputs to a D-dimensional latent space, yielding feature maps $\boldsymbol{z}_i = \mathrm{Enc}(\boldsymbol{x}_i)$. For clarity we assume spatial size $\rho = \zeta = 1$ below.

3.1. Point-Based Prototypes

Euclidean. ProtoPNet [1] computes similarity via the inverted L_2 distance:

$$s_{L_2}(\boldsymbol{z}|\boldsymbol{p}) = \log\left(\frac{\|\boldsymbol{z}-\boldsymbol{p}\|_2^2 + 1}{\|\boldsymbol{z}-\boldsymbol{p}\|_2^2 + \epsilon}\right). \tag{1}$$

This formulation degrades in high dimensions, where L_2 distances lose discriminability.

Cosine. Normalizing vectors to unit length projects them onto a hypersphere, and similarity is measured by the angle between them:

$$s_{\cos}(\boldsymbol{z}|\boldsymbol{p}) = \frac{\boldsymbol{z}^\top \boldsymbol{p}}{\|\boldsymbol{z}\|_2 \|\boldsymbol{p}\|_2} \in [-1, 1]. \tag{2}$$

Hyperspherical geometry benefits classification [8] and avoids the dimensionality issues of L_2.

3.2. Probabilistic Prototypes

Point-based prototypes compare to a cluster center; probabilistic formulations instead model the full cluster distribution, enabling downstream tasks such as outlier detection.

Gaussian. A Gaussian prototype $\boldsymbol{p}^G = (\boldsymbol{\mu}, \boldsymbol{\Sigma})$ defines a multivariate Gaussian over latent space, with similarity given by the PDF:

$$s_{\mathrm{Gauss}}(\boldsymbol{z}|\boldsymbol{p}^G) = \frac{1}{(2\pi)^{\frac{D}{2}} |\boldsymbol{\Sigma}|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(\boldsymbol{z}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{z}-\boldsymbol{\mu})\right). \tag{3}$$

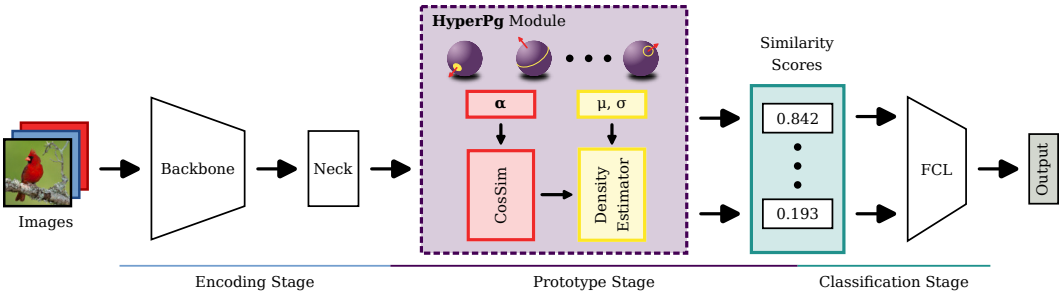


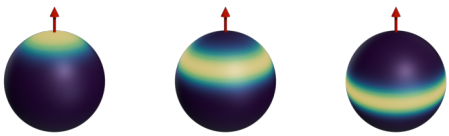
Figure 2. Prototype learning architecture. The prototype module is interchangeable; HyperPG uses a truncated Gaussian PDF as its density estimator.

The covariance $\boldsymbol{\Sigma}$ allows the prototype to adapt its shape to the training data, though learning a full covariance increases computational cost.

HyperPG (ours). We propose *Prototypical Gaussians on the Hypersphere* (HyperPG), a probabilistic prototype for hyperspherical space. A HyperPG prototype $\boldsymbol{p}^H = (\boldsymbol{\alpha}, \mu, \sigma)$ consists of a directional anchor $\boldsymbol{\alpha}$, scalar mean μ, and scalar std σ, and learns a 1D Gaussian over cosine similarities to $\boldsymbol{\alpha}$. Since cosine similarity is bounded to $[-1, 1]$, we use a truncated Gaussian PDF:

$$s_{\text{HyperPG}}(\boldsymbol{z}|\boldsymbol{p}^H) = \frac{\mathcal{N}(s_{\cos}(\boldsymbol{z}|\boldsymbol{\alpha}); \mu, \sigma)}{\mathcal{G}(1, \mu, \sigma) - \mathcal{G}(-1, \mu, \sigma)}, \quad (4)$$

where $\mathcal{G}$ is the Gaussian CDF. As illustrated in Figure 3, varying μ shifts the activation from the anchor pole (μ=1) through ring-shaped patterns ($|\mu|$<1) to the antipodal pole (μ=−1). Crucially, these ring-shaped activations, which would otherwise require an infinite mixture of point prototypes, are captured with just two additional scalar parameters (μ, σ) per prototype, significantly increasing representational power without added computational cost. HyperPG also generalises: setting μ=1 recovers the von Mises-Fisher distribution.



(a) $\mu = 1$ (b) $\mu = 0.5$ (c) $\mu = 0$

Figure 3. HyperPG activations on a 3D hypersphere (anchor $\boldsymbol{\alpha}$=$(0, 0, 1)$, σ=0.1). The mean μ shifts activation from a point at the anchor pole through equatorial rings to the antipodal pole.

4. Training

We adopt the ProtoPNet architecture [1] as our base: a pretrained backbone followed by a two-layer 1×1 convolution neck, a prototype module (interchangeable across formulations), and a linear classification head (Figure 2). Training follows the standard multi-objective loss:

$$L = L_{\text{CE}} + \lambda_{\text{Clst}} L_{\text{Clst}} + \lambda_{\text{Sep}} L_{\text{Sep}},$$

combining cross-entropy with a cluster loss (pulling same-class features toward their prototypes) and separation loss (pushing features away from other-class prototypes), with λ_{Clst}=0.8 and λ_{Sep}=0.08 [1].

Prototypical part networks are known to be hyperparameter-sensitive, requiring complex multi-stage optimization with warm-up phases, stage-specific optimizers, and learning rate schedules [1]. A key question in our experiments is whether hyperspherical formulations reduce this burden: we therefore evaluate all prototype formulations under both the *full* ProtoPNet optimization scheme and a *simplified* single-optimizer training procedure.

5. Experiments

Table 2 reports mean top-1 accuracy and standard deviation over three random seeds. We use ImageNet pretrained weights, without model ensembles or iNaturalist weights.

Full Optimization. Using ProtoPNet's full training scheme [1] with ResNet50 and DenseNet121 backbones on CUB-200-2011 [22], cosine prototypes perform best, outperforming even the blackbox baseline. Gaussian prototypes show high hyperparameter sensitivity and fail entirely with ResNet50.

Simplified Optimization. We replace ProtoPNet's multi-stage optimization with a single AdamW optimizer (lr 1×10^{-4}, no scheduling). Euclidean prototypes suffer a large accuracy drop, revealing that ProtoPNet's gains are partly attributable to its heavily tuned optimization scheme. Hyperspherical formulations remain competitive and converge significantly faster: within 20 epochs versus 200+ for Euclidean prototypes (Figure 4). Results on Stanford Cars [4] and Oxford Flowers [12] confirm this pattern consistently across datasets.

Table 2. Mean Top-1 Test Accuracies (%) CUB-200-2011.

	ProtoPNet Optim.		Simplified Optim.			Add. Datasets (R50)	
Method	R50	D121	R50*	D121*	ViT*	Cars*	Flowers*
Baseline	79.2 ± 0.2	75.5 ± 0.2	79.2 ± 0.2	75.5 ± 0.2	78.5 ± 0.8	85.2	90.0
Euclidean (PPN)	73.9 ± 0.4	76.4 ± 1.2	61.4 ± 1.4	57.8 ± 3.0	23.4 ± 0.4	75.4	70.7
Gaussian	6.4 ± 0.3	75.7 ± 0.2	61.1 ± 0.2	45.9 ± 5.1	4.2 ± 1.3	73.0	74.8
Cosine	$\mathbf{78.5 \pm 0.2}$	$\mathbf{80.4 \pm 0.2}$	71.7 ± 0.4	68.7 ± 0.5	69.2 ± 3.9	**79.6**	85.5
HyperPG (Ours)	77.8 ± 0.2	78.7 ± 0.1	$\mathbf{74.3 \pm 0.6}$	$\mathbf{70.7 \pm 0.3}$	$\mathbf{72.9 \pm 0.1}$	79.0	**87.8**

*Without warm-up epochs and training optimizations. R50: ResNet50, D121: DenseNet121

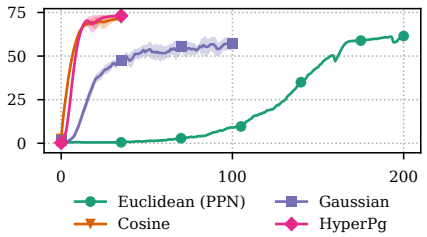
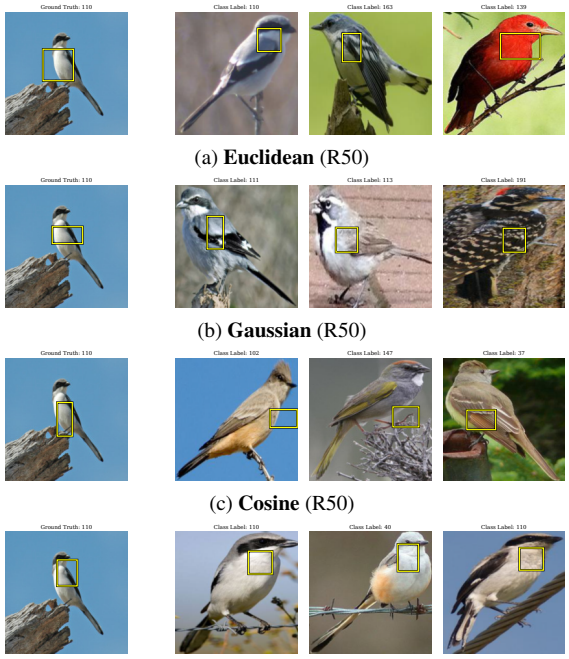


Figure 4. CUB-200-2011 test accuracy per epoch (ResNet50, simplified optimization, mean ± std over 3 seeds). Hyperspherical formulations converge within 20 epochs; Euclidean requires 200+.

Ablation: ViT Backbone. Under the simplified scheme, both Euclidean and Gaussian prototypes fail to learn meaningful representations from ViT-B16 features, while hyperspherical formulations succeed. We hypothesize that the Transformer's attention mechanism induces a near-hyperspherical latent space, making cosine-based prototypes a natural fit. This has direct implications for language-aligned backbones such as CLIP [14], which already use cosine similarity for modality alignment, suggesting hyperspherical prototypes are the only viable formulation in such settings.



(a) **Euclidean** (R50)

(b) **Gaussian** (R50)

(c) **Cosine** (R50)

(d) **HyperPG** (R50)

Figure 5. Left: Test Image. Right: Three closest Training Patches to the Prototype.

6. Interpretability Analysis

Figure 5 shows prototype activation maps (PAMs) for different formulations on the same test image, with the two most activated training patches presented following the prototypical concepts approach [7]. All formulations activate similar regions in the test image, but differ in their associated training examples. Euclidean prototypes produce noticeably less visually consistent associations under the simplified optimization scheme.

7. Conclusion

We present a systematic comparison of point-based and probabilistic prototype formulations in Euclidean and hyperspherical latent spaces, and introduce HyperPG — Gaussian distributions on a hypersphere. Our experiments reveal that hyperspherical prototypes maintain competitive performance under drastically simplified training (single optimizer, no warm-up or scheduling), while Euclidean prototypes degrade severely, suggesting that ProtoPNet's strong performance relies heavily on extensive hyperparameter tuning. Hyperspherical formulations are also the only viable option for ViT and language-aligned backbones such as CLIP. Together, these properties make hyperspherical prototypes a more practical and accessible choice without sacrificing inherent interpretability. Future work could explore adaptive prototype selection or richer probabilistic models such as Mixtures or Bayesian approaches [24].

Acknowledgements

The work was funded by the German Research Foundation (DFG) – 448648559. This work is supported by the Helmholtz Association Initiative and Networking Fund under the KiKIT Pilot Program Core-Informatics. The authors gratefully acknowledge the support of the Robotics Institute Germany (RIG).

References

[1] Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan K. Su. This looks like that: Deep learning for interpretable image recognition. *Neural Information Processing Systems*, 2019. 1, 2, 3

[2] Jonathan Donnelly, A. Barnett, and Chaofan Chen. Deformable protopnet: An interpretable image classifier using deformable prototypes. *Computer Vision and Pattern Recognition*, 2021. 1, 2

[3] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 2

[4] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 554–561, 2013. 1, 3

[5] Oscar Li, Hao Liu, Chaofan Chen, and Cynthia Rudin. Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, New Orleans, Louisiana, USA, 2018. AAAI Press. 2

[6] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems 30*, pages 4765–4774. Curran Associates, Inc., 2017. 1

[7] Chiyu Ma, Brandon Zhao, Chaofan Chen, and Cynthia Rudin. This looks like those: Illuminating prototypical concepts using multiple visualizations. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 4

[8] Pascal Mettes, Elise Van der Pol, and Cees Snoek. Hyperspherical prototype networks. *Advances in neural information processing systems*, 32, 2019. 1, 2

[9] Nazanin Moradinasab, Laura S. Shankman, Rebecca A. Deaton, Gary K. Owens, and Donald E. Brown. Protogmm: Multi-prototype gaussian-mixture-based domain adaptation model for semantic segmentation. *arXiv preprint arXiv:2406.19225*, 2024. 2

[10] Meike Nauta, Ron van Bree, and Christin Seifert. Neural prototype trees for interpretable fine-grained image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14933–14943, 2021. 2

[11] Meike Nauta, Jörg Schlötterer, Maurice van Keulen, and Christin Seifert. Pip-net: Patch-based intuitive prototypes for interpretable image classification. *Computer Vision and Pattern Recognition*, 2023. 2

[12] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Indian Conference on Computer Vision, Graphics and Image Processing*, 2008. 1, 3

[13] Mateusz Pach, Dawid Rymarczyk, Koryna Lewandowska, Jacek Tabor, and Bartosz Zieliński. Lucidppn: Unambiguous prototypical parts network for user-centric interpretable computer vision, 2024. 2

[14] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 4

[15] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?": Explaining the predictions of any classifier, 2016. 1

[16] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature machine intelligence*, 1(5):206–215, 2019. 1

[17] Dawid Rymarczyk, Łukasz Struski, Jacek Tabor, and Bartosz Zieliński. Protopshare: Prototype sharing for interpretable image classification and similarity discovery. *arXiv preprint arXiv:2011.14340*, 2020. 1, 2

[18] Dawid Rymarczyk, Łukasz Struski, Michał Górszczak, Koryna Lewandowska, Jacek Tabor, and Bartosz Zieliński. Interpretable image classification with differentiable prototypes assignment. In *European Conference on Computer Vision*, pages 351–368. Springer, 2022. 2

[19] Mikołaj Sacha, Dawid Rymarczyk, Łukasz Struski, Jacek Tabor, and Bartosz Zieliński. Protoseg: Interpretable semantic segmentation with prototypical parts. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1481–1492, 2023. 1

[20] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017. 1

[21] Y. Ukai, Tsubasa Hirakawa, Takayoshi Yamashita, and H. Fujiyoshi. This looks like it rather than that: Protoknn for similarity-based classifiers. *International Conference on Learning Representations*, 2023. 2

[22] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 1, 3

[23] Chong Wang, Yuyuan Liu, Yuanhong Chen, Fengbei Liu, Yu Tian, Davis J McCarthy, Helen Frazer, and Gustavo Carneiro. Learning support and trivial prototypes for interpretable image classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2062–2072, 2023. 2

[24] Chong Wang, Yuanhong Chen, Fengbei Liu, Davis James McCarthy, Helen Frazer, and Gustavo Carneiro, 2024. 2, 4

[25] Jiaqi Wang, Huafeng Liu, Xinyue Wang, and Liping Jing. Interpretable image recognition by constructing transparent embedding space. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 895–904, 2021. 2

[26] Mengqi Xue, Qihan Huang, Haofei Zhang, Jingwen Hu, Jie Song, Mingli Song, and Canghong Jin. Protopformer: Concentrating on prototypical parts in vision transformers for interpretable image recognition. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*, pages 1516–1524. International Joint Conferences on Artificial Intelligence Organization, 2024. Main Track. 2

[27] Tianfei Zhou, Yi Yang, Ender Konukoğlu, and Luc Van Goo. Rethinking semantic segmentation: A prototype view. *Computer Vision and Pattern Recognition*, 2022. 2