



Interpretable3D: An Ad-Hoc Interpretable Classifier for 3D Point Clouds

Tuo FENG

University of Technology Sydney

Previous Methods & Motivation



Existing explanation studies on 3D models have been conducted with post-hoc explanations. However, post-hoc explanations are problematic and misleading:

- i) Post-hoc analysis requires **a separate modeling effort**, which is not completely faithful to the original model, **with unknown nuances**.
- ii) Post-hoc explanations can **vary depending on** the chosen **explanation models**.
- iii) Post-hoc explanations **cannot provide a reasoning process** for the network's decision-making.
- iv) Post-hoc methods may produce explanations that are **not interpretable to humans**, necessitating extra modeling to ensure understandability.

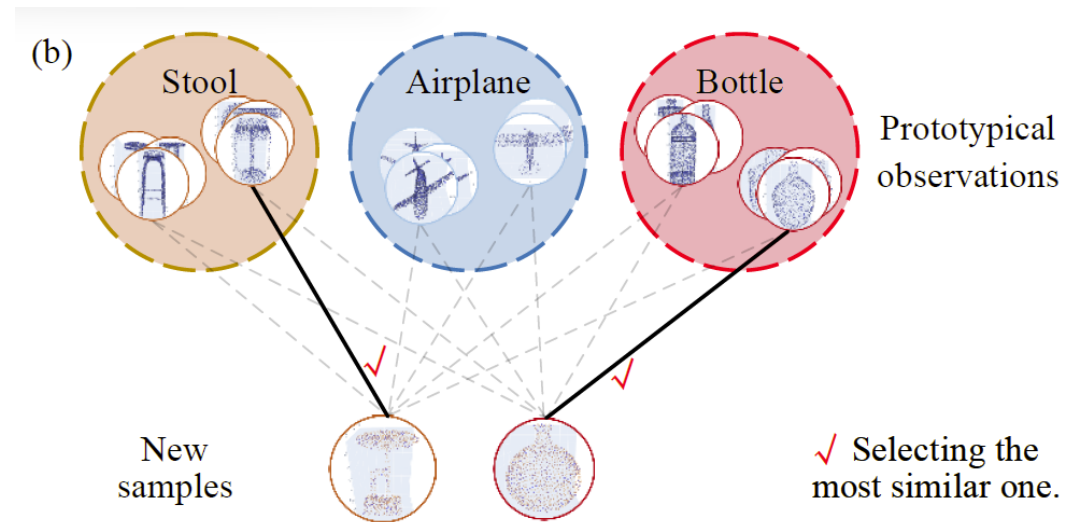
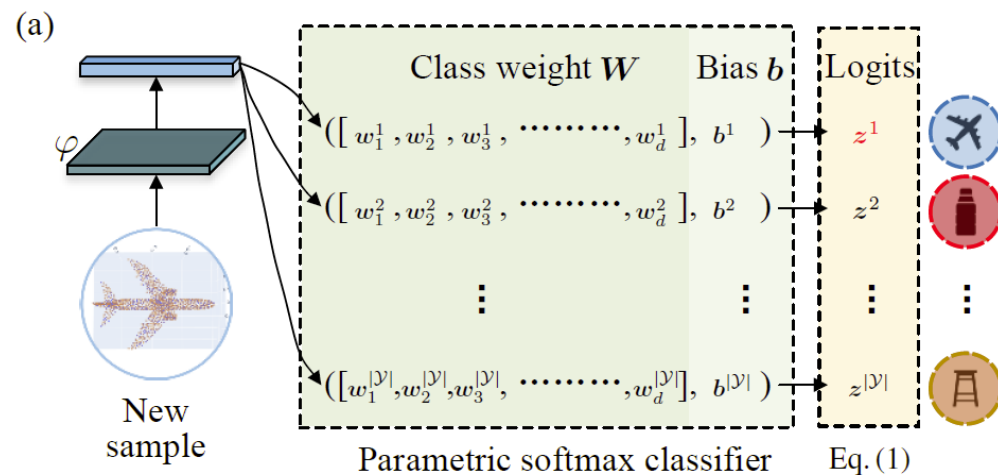
As a result, the demand for interpretable 3D models has become increasingly urgent.

Previous Methods & Motivation

The parametric softmax classifier: 1) learns **highly abstract parameters** that are disconnected from the physical characteristics of the problem being modeled and 2) lacks a **direct and intuitive interpretation**.

To address these issues, we propose an **ad-hoc interpretable classifier** for 3D point clouds.

In Interpretable3D, the prediction is made by assigning the labels of the most similar prototypes to the samples.





An ad-hoc interpretable classifier for 3D point clouds

As an intuitive case-based classifier, Interpretable3D can provide reliable ad-hoc explanations without any embarrassing nuances.

Training:

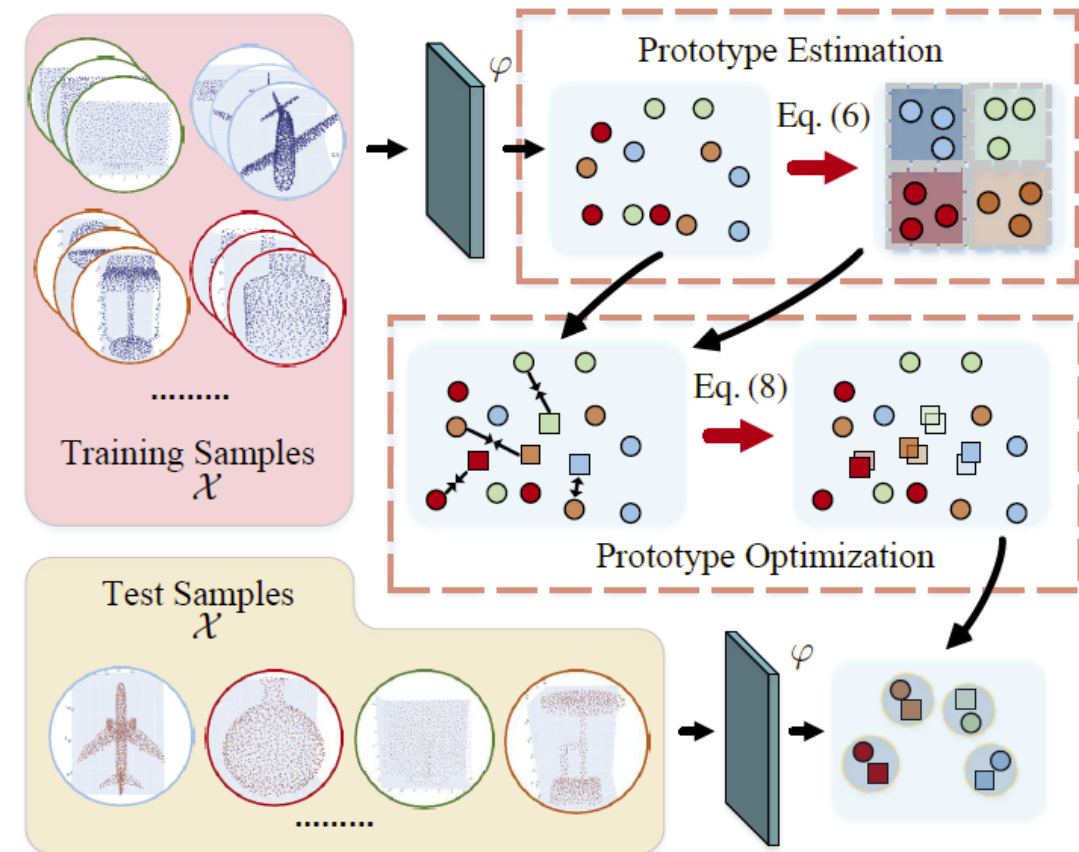
Interpretable3D has two iterative training steps:

- Prototype Estimation
- Prototype Optimization

We update prototypes with their most similar observations in the last few epochs.

Testing:

Interpretable3D classifies new samples according to prototypes.



Prototype Estimation

For class l , we cluster S intra-class representations in the latent space to mine typical prototypes.

Clustering is taken as the *optimal transport problem*.

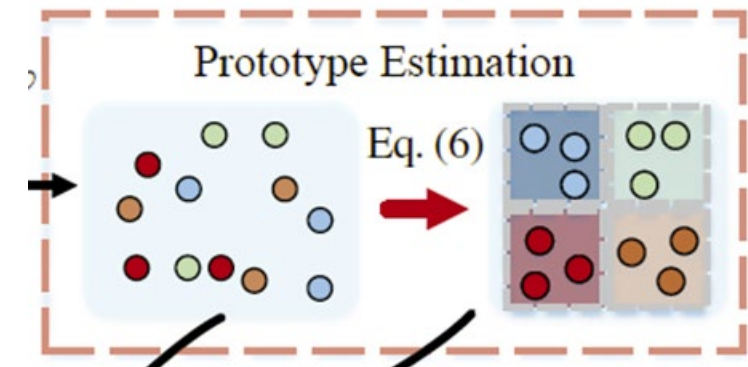
The optimization of assignment matrix A^l is accomplished by optimizing the similarity between the representations F^l and cluster centers M^l , i.e., $Q^l = \text{softmax}(-M^{l\top} F^l)$:

$$A^{l*} = \arg \min_{A^l \geq 0} \langle Q^l, A^l \rangle_F, \quad \text{s.t. } A^l \mathbf{1}_{N^l} = \mathbf{r}, A^{l\top} \mathbf{1}_S = \mathbf{c},$$

The entropic regularization of the above problem can be formulated as:

$$\min_{A^l \geq 0} \langle Q^l, A^l \rangle_F - \zeta H(A^l), \quad \text{s.t. } A^l \mathbf{1}_{N^l} = \mathbf{r}, A^{l\top} \mathbf{1}_S = \mathbf{c}, \zeta > 0,$$

It can be efficiently solved by Sinkhorn-Knopp algorithm.



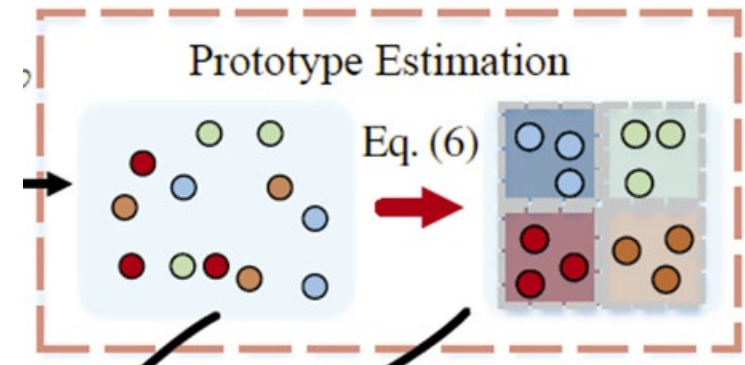
Prototype Estimation

The entropic regularization version is constrained by affine constraints, while the dual problem is unconstrained, making it simpler to design algorithms and analyze complexity.

Therefore, with the Lagrangian function, we further simplify the problem into the dual form:

$$\min_{\mathbf{u} \in \mathbb{R}^S, \mathbf{v} \in \mathbb{R}^{N^l}} \left\{ \mathbf{1}_S^\top \mathbf{A}^{l*} \mathbf{1}_{N^l} - \langle \mathbf{u}, \mathbf{r} \rangle_F - \langle \mathbf{v}, \mathbf{c} \rangle_F \right\},$$

It can be solved by APDAGD.



After completing the label assignment task, we employ a momentum update strategy to update the prototype. It updates each prototype with the average of embeddings of each sub-class center.

However, in the last few epochs, we substitute the average of embeddings with the features of the closest training sample.

Prototype Optimization

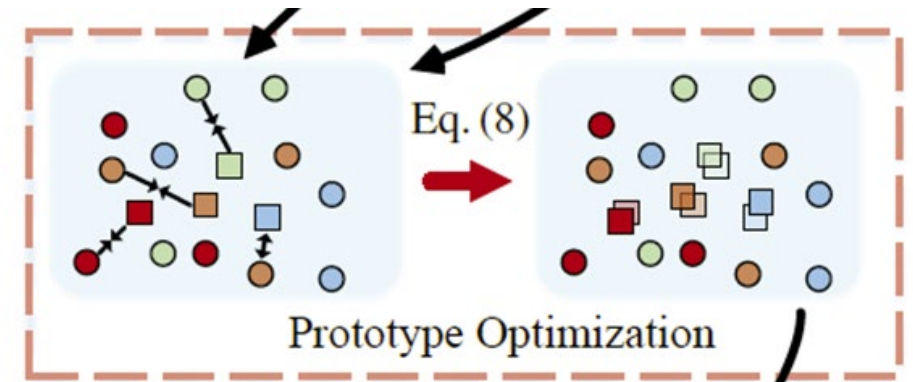
To pursue more representative and discriminative prototypes, we further optimize the prototypes with Prototype Optimization.

We take the estimated prototypes M^l in Prototype Estimation as the initial set.

Only the winning prototypes M^w are altered according to the *competitive learning* update equation (LVQ1):

$$M^w \leftarrow M^w + \eta \psi(l, \hat{l}_w) (F^l - M^w),$$

$$\psi(l, \hat{l}_w) = \begin{cases} +1 & \text{if } l = \hat{l}_w \\ -1 & \text{else} \end{cases},$$



Specifically, if M^w predicts correctly, it is migrated towards the samples' representation F^l . Conversely, M^w is moved away from F^l .

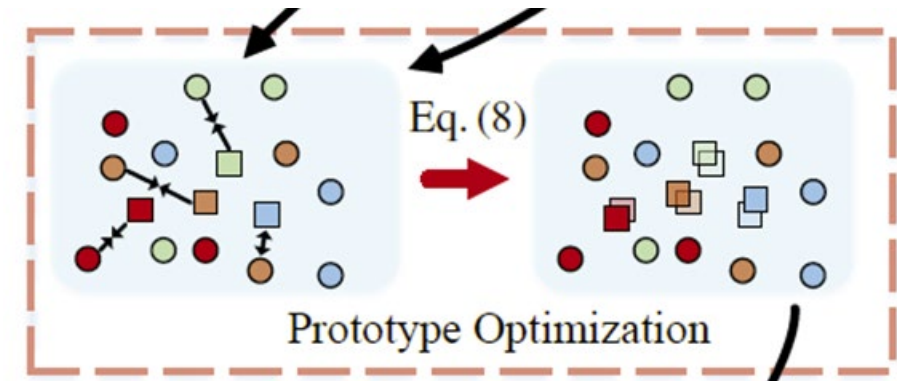
Prototype Optimization

An enhanced LVQ algorithm, known as LVQ2.1, is often favored because it is effective in Bayesian decision theory.

We further investigate LVQ2.1 by updating M^w as follows:

$$\begin{cases} M_p^w \leftarrow M_p^w - \eta (F^l - M_p^w), \\ M_q^w \leftarrow M_q^w + \eta (F^l - M_q^w), \end{cases}$$

$$\text{if } \min \left(\frac{\langle F^l, M_p^w \rangle}{\langle F^l, M_q^w \rangle}, \frac{\langle F^l, M_q^w \rangle}{\langle F^l, M_p^w \rangle} \right) > \frac{1 - \mu}{1 + \mu},$$



Decision boundaries are directly shifted toward the Bayes limits with attractive and repulsive forces from F^l .



Shape Classification on ModelNet40/ScanObjectNN

We evaluate Interpretable3D with four point cloud models: DGCNN, PointNet2, PointMLP, and PointNeXt on two well-known public benchmarks (i.e., ModelNet40 and ScanObjectNN).

Method	Publisher	OA(%)	mAcc(%)
PointNet (Qi et al. 2017a)	CVPR	89.2	86.0
PointNet++ (Qi et al. 2017b)	NeurIPS	90.7	-
PointNet2 (Yan 2019)	—	92.2	-
PointNet2 + Ours		93.2	89.3
DGCNN (Phan et al. 2018)	NN	92.9	90.2
DGCNN + Ours		93.5	90.3
PointMLP (Ma et al. 2021)	ICLR	94.1	91.3
PointMLP + Ours		94.1	92.0
PointNeXt (Qian et al. 2022)	NeurIPS	94.0	91.1
PointNeXt + Ours		94.3	91.8

Table 1: Classification results on ModelNet40 (see §4.1).

Method	Publisher	OA(%)	mAcc(%)
PointNet (Qi et al. 2017a)	CVPR	68.2	63.4
PointNet++ (Qi et al. 2017b)	NeurIPS	77.9	75.4
DGCNN (Phan et al. 2018)	NN	78.1	73.6
DGCNN + Ours		78.0	74.3
PointNet2 (Yan 2019)	—	79.1	77.6
PointNet2 + Ours		79.3	78.4
PointMLP (Ma et al. 2021)	ICLR	85.4	83.9
PointMLP + Ours		85.6	84.5
PointNeXt (Qian et al. 2022)	NeurIPS	87.7	85.8
PointNeXt + Ours		88.0	86.5

Table 2: Classification results on ScanObjectNN (see §4.2).

Interpretable3D achieves **comparable or even better performance** than softmax-based black-box models.

Interpretability - Understanding Prototypes and Data Distribution



Model	OA(%) \uparrow	mAcc(%) \uparrow	MMD2 \downarrow	Witness $_{max}\downarrow$
ProtoPNet	92.83	89.70	0.243	0.577
PointNeXt _{PE}	94.21	91.27	0.101	0.328
PointNeXt _{PO}	94.29	91.77	0.085	0.225

- ❑ PointNeXt_{PO} and PointNeXt_{PE} surpass ProtoPNet with significantly fewer prototypes (600 vs. 6000); **our method learns more representative prototypes.**
- ❑ The better performance of PointNeXt_{PO} can be attributed to the fact that **the LVQ-type algorithm optimizes the distribution of prototypes.**



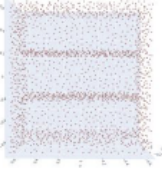
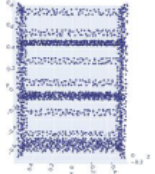
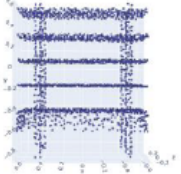
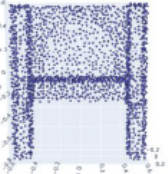
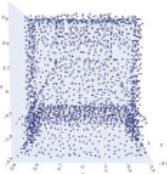
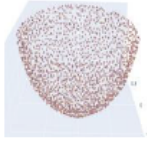
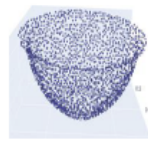
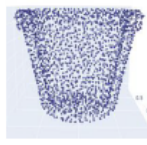
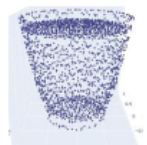
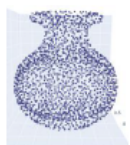
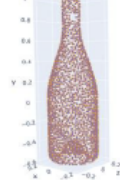
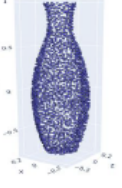
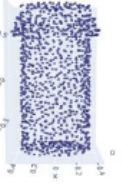
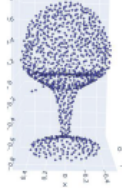
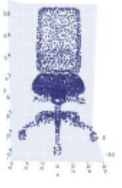
Interpretability for Predictions and Prototypes

Correct cases

ModelNet40

This figure shows how Interpretable3D makes decisions on ModelNet40.

The decision-making mode is straightforward for users.

Test Sample	Prototype 1	Prototype 2	Prototype 3	Prototype 4
				
Bookshelf	Bookshelf	Bookshelf	Mantel	Range Hood
Similarity:	0.8266	0.0971	0.0364	0.0133
				
Bowl	Bowl	Flower Pot	Bowl	Vase
Similarity:	0.8668	0.0583	0.0376	0.0313
				
Bottle	Flower Pot	Bottle	Stool	Chair
Similarity:	0.4263	0.3922	0.0758	0.0526

Failure case

Thanks for Watching

Tuo FENG
University of Technology Sydney