

Backpropagation-Free adaptation of Vision Transformers for Robust and efficient 3D and 2D Perception

by

Moslem YAZDANPANA

MANUSCRIPT-BASED THESIS PRESENTED TO ÉCOLE DE
TECHNOLOGIE SUPÉRIEURE
IN PARTIAL FULFILLMENT FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
Ph.D.

MONTREAL, MARCH 31, 2026

ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC



Moslem Yazdanpanah, 2026



This Creative Commons license allows readers to download this work and share it with others as long as the author is credited. The content of this work cannot be modified in any way or used commercially.

BOARD OF EXAMINERS

THIS THESIS HAS BEEN EVALUATED

BY THE FOLLOWING BOARD OF EXAMINERS

Mr. Christian Desrosiers, Thesis supervisor
Department of Software Engineering and IT, École de Technologie Supérieure

Mr. Ismail Ben Ayed, Thesis Co-Supervisor
Department of Systems Engineering, École de Technologie Supérieure

Mr. Mohamad Forouzanfar, Chair, Board of Examiners
Department of Systems Engineering, École de Technologie Supérieure

Mr. Marco Pedersoli, Member of the Jury
Department of Systems Engineering, École de Technologie Supérieure

Mr. Guillaume-Alexandre Bilodeau, External Independent Examiner
Department of Computer Engineering and Software Engineering, Polytechnique Montréal

THIS THESIS WAS PRESENTED AND DEFENDED

IN THE PRESENCE OF A BOARD OF EXAMINERS AND THE PUBLIC

ON MARCH 13, 2026

AT ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

ACKNOWLEDGEMENTS

Pursuing this Ph.D. has been a formative chapter of my life. It has sharpened my professional abilities, deepened my understanding of deep learning, and broadened my perspective on how research can translate into practice. I am grateful for the intellectual freedom, rigor, and resilience this program cultivated in me.

I owe my deepest gratitude to my supervisor, Prof. Christian Desrosiers, for his guidance, patience, and uncompromising standards. Christian's clarity of thought and constructive feedback shaped the way I approach problems, write, and evaluate ideas. I have learned immensely from his mentorship, technically, scientifically, and personally. I am profoundly thankful for his trust and support. I am also very grateful to my co-supervisor, Prof. Ismail Ben Ayed, whose insights and critical questions consistently strengthened the work.

This journey would not have been possible without my family. To my wife, Ziba, thank you for your unwavering support, patience, and encouragement through every phase, from migration to Canada to the many late nights and long stretches of absence. Your strength and belief carried me forward. To my sons, Aras and Rasa, thank you for your love, joy, and understanding; you reminded me what truly matters and gave meaning to every milestone.

I am grateful to our industrial partner, Zebra Technologies, for the collaboration and for providing practical perspectives that helped anchor the research in real-world constraints. These interactions enriched the work and broadened its impact.

I also thank my teammates and co-authors, Ali Bahri, Mehrdad Noori, Sahar Dastani, David Osowiechi, and Gustavo Adolfo Vargas Hakim, for their collegiality and contributions. A special thanks to Ali, with whom I had a close and often intense collaboration; our discussions, experiments, and iterations were instrumental to several of the ideas in this thesis. I am equally appreciative of the supportive research environment fostered by my lab colleagues and the broader community around our projects.

Finally, to everyone who offered feedback, shared a dataset, debugged code with me, or challenged an assumption, thank you. This thesis reflects not only my efforts but also the generosity and dedication of all those around me.

Adaptation sans rétropropagation des Vision Transformers pour une perception 3D et 2D robuste et efficace

Moslem YAZDANPANAHI

RÉSUMÉ

Les modèles d'apprentissage profond ont obtenu d'excellents résultats en vision par ordinateur, en particulier avec l'arrivée des Transformers et des grands modèles de segmentation. Cependant, ces modèles sont souvent entraînés dans des conditions contrôlées et supposent que les données d'entraînement et de test suivent la même distribution. Dans la pratique, cette hypothèse est rarement vérifiée, surtout dans les applications industrielles en 3D, où les capteurs, les conditions de capture et les objets peuvent fortement varier. Ces décalages de domaine entraînent des chutes de performance importantes et limitent la fiabilité des systèmes en production.

Cette thèse se concentre sur l'Adaptation en Temps de Test (Test-Time Adaptation, TTA) dans des scénarios concrets de vision 3D et d'objets industriels. Nous considérons deux cadres principaux. Le premier concerne la classification robuste de nuages de points avec des Transformers 3D soumis à des corruptions et à des changements de domaine. Le second porte sur la segmentation d'instances d'objets « modèle-basés » jamais vus à l'entraînement, mais décrits par des modèles CAO ou quelques vues de référence (scénario MISUO). Dans ces deux contextes, nous faisons face à des contraintes fortes : absence de labels en phase de test, impossibilité d'accéder aux données sources, et besoin de méthodes légères en temps de calcul et en mémoire.

Notre première contribution introduit *Purge-Gate*, une méthode d'adaptation en temps de test sans rétropropagation pour les Transformers sur nuages de points. L'idée centrale est de travailler au niveau des tokens. Nous estimons, pour chaque token, à quel point il semble « déplacé » par rapport à la distribution source, puis nous supprimons les tokens les plus affectés avant les couches d'attention. Ainsi, le modèle concentre son attention sur des régions plus fiables du nuage de points. Nous proposons deux variantes : PG-SP, qui exploite des statistiques de la source lorsque celles-ci sont disponibles, et PG-SF, qui est entièrement sans source et utilise un prototype dérivé du token CLS. Ces deux variantes fonctionnent sans gradients en phase de test et améliorent nettement la robustesse, la vitesse et l'efficacité mémoire sur plusieurs benchmarks 3D corrompus.

Dans notre deuxième contribution, nous revisitons le rôle de la Layer Normalization (LN) pour l'adaptation en temps de test des Transformers 3D. Nous montrons théoriquement que, lorsque la dimension des caractéristiques et la taille de lot augmentent, la sortie pré-affine de LN se concentre autour d'une moyenne nulle et d'une variance unitaire. Dans ce régime, ré-estimer des statistiques de lot à la manière de BatchNorm n'est pas nécessaire et peut même dégrader les performances. À partir de cette analyse, nous proposons *LN-TTA*, une méthode simple qui laisse intacte la partie pré-affine de LN et n'ajuste que les paramètres d'affinité (biais et gain) en une seule passe avant. LN-TTA est sans source, sans rétropropagation, très légère en calcul, et apporte des gains significatifs sur plusieurs suites de nuages de points corrompus.

Enfin, notre troisième contribution étend ces idées à un cadre plus complexe : la segmentation d’instances d’objets modèle-basés jamais vus à l’entraînement. Nous présentons *PRISMA*, un cadre sans entraînement qui affine les prompts de grands modèles de segmentation à partir de prototypes visuels construits à partir de modèles CAO ou d’images multi-vues. *PRISMA* s’appuie sur trois modules complémentaires : (i) un module de génération de propositions de masques (MPG) qui utilise des descripteurs de type DINO pour proposer des régions d’intérêt adaptées aux objets, (ii) un module d’auto-ancrage (Self-Grounding) qui injecte les prototypes d’objets dans le backbone et sépare tokens de premier plan et d’arrière-plan via une attention bloc-diagonale, sans réentraînement, et (iii) un schéma de pondération par la confiance (CW) qui exploite les scores internes du segmentateur pour mieux classer les prédictions. Sur le benchmark BOP, *PRISMA* améliore la qualité et la rapidité de la segmentation en MISUO par rapport aux méthodes existantes, tout en restant complètement sans entraînement.

Dans l’ensemble, cette thèse montre qu’il est possible de rendre des modèles modernes de vision 2D/3D plus robustes et plus adaptés aux contraintes industrielles en s’appuyant sur des mécanismes d’adaptation en temps de test simples, analytiques et sans rétropropagation. En travaillant directement sur la normalisation, la sélection de tokens et l’exploitation de prototypes visuels, nous proposons une nouvelle boîte à outils pour déployer des modèles puissants dans des environnements 3D réels, sans nécessiter de coûteux processus de réentraînement.

Mots-clés: Adaptation en Temps de Test, Sans Rétropropagation, Vision Transformers, Nuages de Points 3D, Segmentation d’Objets Modèle-Basés, Robustesse aux Décalages de Domaine, Perception Industrielle

Backpropagation-Free adaptation of Vision Transformers for Robust and efficient 3D and 2D Perception

Moslem YAZDANPANAHI

ABSTRACT

Deep learning models have achieved excellent results in computer vision, especially with the advent of Transformers and large segmentation models. However, these models are often trained under controlled conditions and assume that the training and test data follow the same distribution. In practice, this assumption is rarely satisfied, especially in 3D industrial applications, where sensors, capture conditions, and objects can vary greatly. These domain shifts lead to significant drops in performance and limit the reliability of systems in production.

This thesis focuses on Test-Time Adaptation (TTA) in concrete 3D vision and industrial object scenarios. We consider two main settings. The first concerns robust classification of point clouds with 3D Transformers subject to corruptions and domain shifts. The second addresses instance segmentation of “model-based” objects that are never seen during training but are described by CAD models or a few reference views (the MISUO scenario). In both contexts, we face strong constraints: the absence of labels at test time, the impossibility of accessing source data, and the need for methods that are lightweight in terms of computation time and memory.

Our first contribution introduces *Purge-Gate*, a backpropagation-free test-time adaptation method for Transformers on point clouds. The core idea is to operate at the token level. For each token, we estimate how “shifted” it appears with respect to the source distribution, and then remove the most affected tokens before the attention layers. This allows the model to focus its attention on more reliable regions of the point cloud. We propose two variants: PG-SP, which uses source statistics when they are available, and PG-SF, which is completely source-free and uses a prototype derived from the CLS token. Both variants operate without gradients at test time and significantly improve robustness, speed, and memory efficiency on several corrupted 3D benchmarks.

In our second contribution, we revisit the role of Layer Normalization (LN) for test-time adaptation of 3D Transformers. We show theoretically that, as the feature dimension and batch size increase, the pre-affine output of LN concentrates around zero mean and unit variance. In this regime, re-estimating batch statistics in the style of BatchNorm is unnecessary and can even degrade performance. Based on this analysis, we propose *LN-TTA*, a simple method that keeps the pre-affine part of LN unchanged and adjusts only the affine parameters (bias and gain) in a single forward pass. LN-TTA is source-free, backpropagation-free, very lightweight computationally, and yields significant gains on several corrupted point cloud suites.

Finally, our third contribution extends these ideas to a more complex setting: instance segmentation of model-based objects that are never seen during training. We present *PRISMA*, a training-free framework that refines prompts for large segmentation models using visual prototypes built from CAD models or multi-view images. PRISMA relies on three complementary

modules: (i) a Mask Proposal Generation (MPG) module that uses DINO-style descriptors to propose object-specific regions of interest, (ii) a Self-Grounding module that injects object prototypes into the backbone and separates foreground and background tokens through block-diagonal attention, without retraining, and (iii) a Confidence-Weighted (CW) scheme that leverages the segmenter’s internal scores to better rank predictions. On the BOP benchmark, PRISMA improves both the quality and the speed of MISUO segmentation compared to existing methods, while remaining completely training-free.

Overall, this thesis shows that it is possible to make modern 2D/3D vision models more robust and better suited to industrial constraints by relying on simple, analytical, backpropagation-free test-time adaptation mechanisms. By acting directly on normalization, token selection, and the use of visual prototypes, we propose a new toolbox for deploying powerful models in real 3D environments without requiring costly retraining procedures.

Keywords: Test-Time Adaptation, Backpropagation-Free, Vision Transformers, 3D Point Clouds, Model-Based Unseen Object Segmentation, Domain Shift Robustness, Industrial Perception

TABLE OF CONTENTS

	Page
INTRODUCTION	1
0.1 Research statement	4
0.2 Contributions	5
0.2.1 Additional contributions	7
0.2.2 External collaboration	7
CHAPTER 1 LITERATURE REVIEW	9
1.1 Introduction	9
1.2 From Domain Shift to Test-Time Adaptation	11
1.2.1 Origins in Domain Adaptation	11
1.2.2 The Emergence of Test-Time Training and Adaptation	13
1.3 Architectural Foundations for Adaptation	15
1.3.1 Normalization as a Vehicle for Adaptation	15
1.3.2 Object-centric point cloud classification	16
1.3.3 Transformers and Tokenization in 3D Vision	17
1.4 Test-Time Adaptation in 3D Perception	19
1.4.1 Challenges of Test-Time Shift in Point Clouds	19
1.4.2 From BN-Based 2D TTA to Token-Level Adaptation in 3D Transformers	20
1.4.3 Baseline details	20
1.4.4 The Shift Toward Gradient-Free Adaptation	22
1.4.5 Normalization-Based Adaptation	23
1.4.5.1 BN-based adaptation and its limitation under architectures with LN.	24
1.4.5.2 Why an LN-native solution is needed.	24
1.4.5.3 Our answer: LN-TTA.	25
1.5 Adaptation-Free Segmentation and Prompt Refinement: The PRISMA Framework	25
1.5.1 Baselines details	27
1.5.1.1 What is missing.	29
1.5.1.2 Positioning within the thesis’ forward-only paradigm.	29
1.6 Benchmarking and Evaluation Protocols	29
1.6.1 Test-Time Point-Cloud Classification	30
1.6.1.1 Benchmarks and corruption taxonomies.	30
1.6.1.2 Backbones and pretraining.	30
1.6.1.3 Problem setting and protocols.	30
1.6.1.4 Baselines by adaptation mechanism.	31
1.6.1.5 Metrics and reporting.	31
1.6.2 Model-Based Instance Segmentation of Unseen Objects (MISUO)	32
1.6.2.1 Benchmark: BOP and datasets.	32
1.6.2.2 Tasks and evaluation metrics.	32
1.6.2.3 Method families and inputs.	32

1.6.2.4	Reporting conventions and fairness.	33
1.7	Concluding Remarks	33
CHAPTER 2 PURGE- GATE: BACKPROPAGATION-FREE TEST-TIME ADAPTATION FOR POINT CLOUDS CLASSIFICATION VIA TOKEN PURGING 35		
2.1	Introduction	35
2.2	Related Works	38
2.3	Method	40
2.3.1	Attention Under Distribution Shift	41
2.3.2	Purge-Gate (PG)	44
2.3.2.1	Source-Prototype Purge-Gate (PG-SP)	44
2.3.2.2	Source-Free Purge-Gate (PG-SF)	45
2.3.3	Backpropagation-Free Considerations	47
2.3.3.1	Test-Time Hyperparameter Selection	47
2.3.3.2	Normalization Statistics Update	47
2.4	Experiments	48
2.4.1	Implementation Details	48
2.4.2	Datasets	49
2.4.3	Main Results	49
2.4.4	Ablation Study	52
2.5	Conclusions	54
CHAPTER 3 REVISITING LAYER NORMALIZATION FOR POINT CLOUD TEST TIME ADAPTATION		
3.1	Introduction	55
3.2	Related Works	58
3.3	Method	60
3.3.1	Preliminaries	61
3.3.2	Geometric (Projection) View of LN	63
3.3.3	Effect on Mean and Variance: Token-wise Identities and Batch-wise Limits	64
3.3.4	Backpropagation-Free LN-TTA	65
3.4	Experiments	69
3.4.1	Results	70
3.4.1.1	LN Analyses	70
3.4.1.2	TTA Evaluation	71
3.5	Conclusion	73
CHAPTER 4 PRISMA: PROMPT REFINEMENT FOR INSTANCE SEGMENTATION OF MODEL-BASED UNSEEN OBJECTS WITH ADAPTATION-FREE SELF-GROUNDING		
4.1	Introduction	75
4.2	Related Works	78

4.3	Method	81
4.3.1	Task Definition	81
4.3.2	Preliminaries	82
4.3.2.1	Object-Model Representation	82
4.3.3	Model-Based Prompt Generation (MPG)	83
4.3.4	Self-Grounding	84
4.3.5	Confidence-Weighted IoU Scoring (CW)	86
4.4	Experiments	86
4.4.1	Model-Based Instance Segmentation of Unseen (MISUO)	87
4.4.2	6DOF Pose Estimation of Unseen Objects	88
4.4.3	Model-Based 2D Detection of Unseen Objects	89
4.4.4	Ablation Studies and Analyses	90
4.5	Conclusion	93
CONCLUSION AND RECOMMENDATIONS		95
APPENDIX I SUPPLEMENTARY MATERIAL FOR THE PAPER TITLED PURGE- GATE: BACKPROPAGATION-FREE TEST-TIME ADAPTATION FOR POINT CLOUD CLASSIFICATION VIA TOKEN PURGING		99
APPENDIX II SUPPLEMENTARY MATERIAL FOR THE PAPER TITLED REVISITING LAYER NORMALIZATION FOR POINT CLOUD TEST TIME ADAPTATION		105
APPENDIX III SUPPLEMENTARY MATERIAL FOR THE PAPER TITLED PRISMA: PROMPT REFINEMENT FOR INSTANCE SEGMENTATION OF MODEL-BASED UNSEEN OBJECTS		117
BIBLIOGRAPHY		129

LIST OF TABLES

	Page
Table 1.1	Comparison of <i>Domain Adaptation</i> (DA), <i>Domain Generalization</i> (DG), and <i>Test-Time Adaptation</i> (TTA) along practical, statistical, and deployment axes. Examples reference canonical methods from the literature review (see citations) 12
Table 2.1	Top-1 Classification Accuracy (%) for all distribution shifts in the ScanObjectNN-C dataset. The mean improvements over Source Only are shown in parentheses. Methods with * are reproduced. TTT methods' results are taken from (Bahri <i>et al.</i> , 2025a) 50
Table 2.2	Top-1 Classification Accuracy (%) for all distribution shifts in the ModelNet-C dataset. The mean improvements over Source Only are shown in parentheses. Methods with * are reproduced. TTT methods' results are taken from (Bahri <i>et al.</i> , 2025a) 50
Table 2.3	Top-1 Classification Accuracy (%) for all distribution shifts in the ShapeNet-C dataset. The mean improvements over Source Only are shown in parentheses. Methods with * are reproduced. TTT methods' results are taken from (Bahri <i>et al.</i> , 2025a) 52
Table 2.4	Top-1 Classification Accuracy (%) on Mamba networks for all distribution shifts in ScanObjectNN-C, ModelNet-C, and ShapeNet-C datasets. The mean improvements over Source Only are shown in parentheses. The Mamba backbone is from (Liang <i>et al.</i> , 2024) 52
Table 3.1	Top-1 Classification Accuracy (%) for all distribution shifts in the ScanObjectNN-C dataset. The mean improvements over Source Only are shown in parentheses. Methods with ● use backpropagation during the adaptation. Methods with ● require accessing the source (training-time) data. Point Cloud Per Second (PPS) represents the method's throughput. Bold values in standard mode show the best values among the Backpropagation-Free methods. Results of methods with * are reproduced 69
Table 3.2	Top-1 Classification Accuracy (%) for all distribution shifts in the ModelNet-C dataset. The mean improvements over Source Only are shown in parentheses. Methods with ● use backpropagation during the adaptation. Methods with ● require accessing the source (training-time) data. Point Cloud Per Second (PPS) represents the method's throughput. Bold values in standard mode show the best values among

	the Backpropagation-Free methods. Results of methods with * are reproduced	69
Table 3.3	Top-1 Classification Accuracy (%) for all distribution shifts in the ShapeNet-C dataset. The mean improvements over Source Only are shown in parentheses. Methods with ● use backpropagation during the adaptation. Methods with ● require accessing the source (training-time) data. Point Cloud Per Second (PPS) represents the method's throughput. Bold values in standard mode show the best values among the Backpropagation-Free methods. Results of methods with * are reproduced	70
Table 4.1	Evaluation of our method in BOP benchmark (Hodan <i>et al.</i> , 2024), the Model-Based Instance Segmentation of Unseen (MISUO) task. Methods with superscript ^f use FastSAM for segmentation, and the rest use SAM. In our method, SG stands for Self-Grounding and CW stands for confidence weighting. Results for methods with ● (baseline) are reproduced on our hardware to have a fair time comparison. ● showcases the methods with test-time adaptation. The time for NIDS-Net + CA, and NIDS-Net + WA is unknown due to the unknown adaptation time. GD in Extra Backbones consists of Clip (Radford <i>et al.</i> , 2021), and DINO (Zhang <i>et al.</i> , 2022a). Reported numbers are Average Precision	88
Table 4.2	Evaluation of the effect of our method on downstream 6DoF pose estimation of unseen objects. Methods with superscript ^f use FastSAM for segmentation. In FoundationPose (Wen, Yang, Kautz & Birchfield, 2024), GT showcases corrupted ground-truth segmentation masks. In FoundPose (Örnek <i>et al.</i> , 2025), letters C and F represent Coarse and Feature-refinement respectively. Results for methods with ● are reproduced on our hardware to have a fair time comparison. Reported numbers are Average Recall	89
Table 4.3	Evaluation of our method in BOP benchmark (Hodan <i>et al.</i> , 2024), the task of Model-Based 2D Detection of Unseen Objects . The segmentation model is SAM for all methods. In our method, SG stands for Self-Grounding and CW stands for confidence weighting. ● showcase the methods with test-time adaptation. The time for NIDS-Net + WA is unknown due to the unknown adaptation time. GD in Extra Backbones consists of Clip (Radford <i>et al.</i> , 2021), and DINO (Zhang <i>et al.</i> , 2022a). Reported numbers are Average Precision	90

LIST OF FIGURES

	Page
Figure 1.1	The overall schematic of TTT methods. During the pretraining, the backbone and its dual heads get updated. During the Test-Time Training, just the main backbone and self-supervised head get updated .. 13
Figure 1.2	In Test-Time Adaptation, given a pretrained network with the task head, we only update a small subset of the network, using a label-agnostic objective 14
Figure 1.3	Canonical point-set learning pipeline (PointNet-style): shared pointwise embedding, symmetric aggregation (max pooling) to form a global descriptor, optional alignment modules (T-Net), and prediction heads for classification and per-point segmentation 17
Figure 1.4	Compared to a 2D image, a point cloud is an unorganized and unordered set of 3D points of arbitrary size 18
Figure 1.5	The process of tokenization of 3D point clouds into embedded tokens. These tokens are fed into Transformer layer for further processing 18
Figure 1.6	In token pruning, the goal is to remove the background and unimportant tokens to reduce the memory and computation footprint of transformer networks. Adapted from (Wu, Zeng, Wang & Chen, 2023) 22
Figure 1.7	Comparison of BatchNorm and LayerNorm as adaptation interfaces. BatchNorm couples samples through batch moments, enabling moment swapping at test time. LayerNorm normalizes within each token across features, with no batch coupling, so there is no persistent domain axis to realign 24
Figure 1.8	The MISUO, pipeline from ZeroPose to SAM6D. (a) In this pipeline, the object’s model is utilized to detect the instance-based segmentation of all instances of objects of interest (foreground objects). (b) , Former methods (ZeroPose, CNOS, SAM6D) have focused on improving the scoring techniques, neglecting the effect of over-prompting on the algorithm’s efficiency 26
Figure 1.9	Grounded-SAM utilizes Grounding-DINO, and a text prompt to find the instance Bounding Boxes (BBBox). In industrial settings, due to the MMP issue, objects can not clearly be defined using text prompts 26

Figure 2.1	Token-level domain shift visualization. From left to right: clean point cloud, corrupted point cloud, tokens ranked by PG-SF, and tokens ranked by PG-SP. The color gradient from red to green indicates the degree of domain shift, with red representing the highest shift and green the lowest 36
Figure 2.2	The overall procedure of the PG algorithm at the input of the attention layer 40
Figure 2.3	Effect of purge size on ACC of Background corruption from ScanObjectNN-C dataset 53
Figure 3.1	BN vs. LN: axes and TTA levers. BN normalizes <i>per-feature across the batch</i> , stabilizing per-feature marginals; LN normalizes <i>per-token across features</i> , removing common mode/rescaling while batch-agnostic. These axes yield different test-time levers: BN updates batch statistics, whereas LN minimally recalibrates post-affine (γ, β) . Squares denote activations; colors index samples 56
Figure 3.2	(a) Pre-training. The [cls] token learns to aggregate domain information as it attends to the training-time token stream. (b) Prototype extraction. At test time, with the model frozen, we run a single forward pass using only the pre-trained [cls] token and store layer-wise prototypes for each LN layer. (c) Test-Time Adaptation (TTA). During evaluation on corrupted target data, we reparameterize the <i>post-affine</i> of each LN layer with the stored prototypes, adapting tokens in a single forward pass (no gradients, no source data) 66
Figure 3.3	Average of GPU memory (log scale; lower is better). The reported numbers are for total GPU memory (Backbone + Data) with $B = 32$ 71
Figure 3.4	Jensen–Shannon (JS) distance between the <i>average per-feature</i> post-LN batch marginal and $\mathcal{N}(0, 1)$ over a (B, d) grid. Shown as a contour heat map with $\log_2$ axes. The star marks the (B, d) setting used in our TTA evaluations 72
Figure 4.1	Average Precision (AP) and inference Time of our method compared to other SOTA methods in Model-Based Instance Segmentation of Unseen Objects. The bubble size represents the efficiency (AP penalized by time) 76
Figure 4.2	Comparison of different prompting strategies. Light areas indicate the prompted regions. From left to right: full scene prompting Lin, Liu, Lu & Jia (2024), Grounding-Dino Zhang <i>et al.</i> (2022a), Our Model-Based Prompt Generation (MPG), and Our Self-Grounding

	approach. Grounding-Dino struggles to distinguish individual objects, resulting in inaccurate prompts and insufficient segmentation masks (Maximum of 3 masks). Our MPG module separates the foreground from the background but includes some false positives, while our Self-Grounding module focuses more accurately on foreground areas, significantly reducing false positives. Scene is taken from the HB dataset (Kaskman, Zakharov, Shugurov & Ilic, 2019) 77
Figure 4.3	Examples of similar industrial objects from the T-LESS dataset (Hodan <i>et al.</i> , 2017) (objects 13–16). Due to the MMP, these objects are indistinguishable using language-based descriptors, underscoring the limitations of text-driven prompting 80
Figure 4.4	The notations in this figure follow Section 4.3. a) MISUO pipeline (e.g., SAM6D): SAM receives full-scene prompts to produce candidate masks, which are then labeled and scored (Lin <i>et al.</i> , 2024). SAM’s internal confidence scores are not fully exploited. b) Our MPG module reuses existing backbones and generates a sparse set of foreground prompts for SAM, reducing redundant queries. c) Left: our Self-Grounding (SG) module injects object embeddings into the scene backbone via intra-modality grounding, separating foreground from background and reducing false-positive prompts; Right: SG attention layer details. Skip connections are omitted for clarity. The operator $\llbracket \cdot, \cdot \rrbracket$ denotes position-aware concatenation 81
Figure 4.5	Prompting strategies for MISUO. Left: full-scene prompting (Lin <i>et al.</i> , 2024) queries every region, producing exhaustive but inefficient proposals. Second: text-based Grounding-DINO (Zhang <i>et al.</i> , 2022a) often fails in multi-model product (MMP) settings, merging or missing objects. Third: our MPG focuses prompts on object-like regions, reducing redundancy but leaving some false positives. Right: Self-Grounding (SG) further refines prompting by re-routing attention in the backbone, sharply focusing on true foreground areas. Example from HB (Kaskman <i>et al.</i> , 2019) 84
Figure 4.6	Effect of different similarity thresholds τ_+ on run-time (Top) and accuracy (Bottom) of our method over LM-O dataset (Brachmann <i>et al.</i> , 2014). The broken line shows the threshold we have used in our experiments 91
Figure 4.7	Effect of Confidence-Weighted Scoring (CW) on SAM6D and different components of our method, MPG, and SG 92
Figure 4.8	Ablation of τ_G on time and accuracy, on LM-O dataset 93

LIST OF ABBREVIATIONS AND ACRONYMS

6DoF	Six Degrees of Freedom
BN	Batch Normalization
CW	Confidence-Weighted Scoring
FPS	Farthest Point Sampling
LLN	Law of Large Numbers
LN	Layer Normalization
LN-TTA	LN Test Time Adaptation
MISUO	Model-Based Instance Segmentation of Unseen
MMP	Modality Mismatch Problem
MPG	Model-Based Prompt Generation
PG	Purge-Gate
PG-SF	Source-Free Purge-Gate
PG-SP	Source-Prototype based Purge-Gate
SAM	Segment Anything Model
SFDA	Source-Free Domain Adaptation
SG	Self-Grounding
SOTA	State Of The Art
TTA	Test-Time Adaptation
TTT	Test-Time Training
UDA	Unsupervised Domain Adaptation

LIST OF SYMBOLS AND UNITS OF MEASUREMENTS

X_i	Raw point-cloud sample i , a set of 3D points $X_i \subset \mathbb{R}^3$
$p_{i,n}$	n -th 3D point of sample i
N_p	Number of points in a point cloud
x_i	Token matrix of sample i , $x_i \in \mathbb{R}^{N \times d}$
$x_{i,t}$	t -th token of sample i , $x_{i,t} \in \mathbb{R}^d$
N	Number of tokens per point cloud (or per sequence)
d	Feature (embedding) dimension
$\mathcal{D}_S$	Source dataset $\{(x_i^S, y_i)\}_{i=1}^{N_S}$
$\mathcal{D}_T$	Target dataset $\{x_j^T\}_{j=1}^{N_T}$
N_S, N_T	Number of source and target samples
$P_S(x), P_T(x)$	Source and target data distributions in embedding space
y_i	Class label of sample i
$\hat{y}_i$	Predicted logits for sample i
f_θ	Transformer-based classifier with parameters θ
$\mathcal{E}$	Token embedding map from local neighborhoods to $\mathbb{R}^d$
B	Test-time batch size (number of sequences per batch)
c_t	Center point of token t chosen by farthest point sampling
$\mathcal{N}_t$	Local neighborhood (KNN patch) around c_t
$\text{KNN}_k(\cdot; X)$	k -nearest-neighbors operator on point set X

$\ \cdot\ _2$	Euclidean (ℓ_2) norm
$\text{LN}(x_t)$	LayerNorm applied to token x_t
$\widehat{\text{LN}}(x_t)$	Pre-affine LN output (projection-and-rescaling)
γ, β	LN affine parameters, $\gamma, \beta \in \mathbb{R}^d$
Γ	Diagonal matrix $\Gamma = \text{diag}(\gamma)$
$\mu_d(x_t)$	Mean of token x_t across its d features
$\sigma_d^2(x_t)$	Variance of token x_t across its d features
ε	LN stability constant (epsilon)
$\mathbf{1}$	All-ones vector in $\mathbb{R}^d$
$\mathbf{u}$	Normalized common-mode direction, $\mathbf{u} = \mathbf{1}/\sqrt{d}$
$\mathbf{C}$	Centering projector $I - \mathbf{u}\mathbf{u}^\top$ onto mean-zero subspace
$\mathcal{H}$	Mean-zero hyperplane $\{v : \mathbf{u}^\top v = 0\}$
θ	Direction of mean-free token, $\theta \in \mathcal{H} \cap \mathbb{S}^{d-2}$
r	Mean-free energy $r = d^{-1} \ \mathbf{C}x_t\ _2^2$ in LN scale analysis
$f(a; x_t)$	LN scale factor under global rescaling $x_t \mapsto ax_t$
$H_{b,t}$	Pre-affine LN output for batch element b , token t
$\overline{H}_{j,B}$	Batch mean of feature j across B samples at a fixed token
$S_{j,B}^2$	Batch variance of feature j across B samples
$y_{b,t}$	Post-affine LN output for batch element b , token t
$\mu_\star^S$	Per-feature mean of post-affine LN outputs on source domain

$\Sigma_{\star}^S$	Diagonal covariance of post-affine LN outputs on source domain
γ_0^S, β_0^S	Source-trained LN parameters
$\gamma^{\star}, \beta^{\star}$	Adapted per-feature LN parameters (moment-matched)
γ'^T, β'^T	Scalar-reparameterized LN parameters for target domain
$\hat{\mu}_h^T$	Empirical mean of pre-affine outputs on target window
$(\hat{\sigma}_{h,j}^2)^T$	Empirical variance of pre-affine outputs on target window, feature j
δ	Small ridge term added to variances for numerical stability
$\bar{m}(\beta)$	Average of LN bias parameters, $d^{-1}\mathbf{1}^{\top}\beta$
$\bar{s}(\gamma)$	RMS of LN scales, $(d^{-1}\ \gamma\ _2^2)^{1/2}$
y_{proto}	Post-affine LN output for the stored [cls] prototype
$m_{\star}^{\text{proto}}$	Mean of y_{proto} across features
$s_{\star}^{\text{proto}}$	RMS of y_{proto} across features
L_{pg}	Number of tokens to purge (purging length)
$\mathcal{L}_{\text{pg}}$	Candidate set of purging lengths tested at TTA
L_{pg}^*	Selected purging length minimizing entropy at test time
$\epsilon_{j,t}$	Additive noise perturbation on token (j, t)
G	Prototype used by PG (source or [cls]-based)
$\Delta(x_j^T, G)$	Vector of token-wise divergences from prototype G
$\delta_{j,t}^T$	Divergence score of token t in target sample j
$\hat{x}_j^T$	Purified token sequence after removing top- L_{pg} divergent tokens

$H(z)$	Entropy of logits z used for hyperparameter selection
p_c	Softmax probability for class c given logits z
W_Q, W_K, W_V	Query, Key, and Value projection matrices in attention
Q, K, V	Query, Key, and Value token matrices
$S_{t,t'}$	Attention score between tokens t and t'
$A_{t,t'}$	Attention weight between tokens t and t'
u, v	Random directions on a high-dimensional sphere (used in concentration arguments)
$\mathbb{O}$	Set of objects $\{O_1, \dots, O_{n_O}\}$
n_O	Number of objects
$\mathbb{C}$	Set of camera views $\{C_1, \dots, C_{n_C}\}$
n_C	Number of camera views
$\mathbb{M}_{\mathbb{C}}^{O_o}$	Multi-view images of object O_o captured from cameras $\mathbb{C}$
$\mathbf{F}_O^\ell$	Local object feature tensor, $n_O \times n_C \times H'W' \times d$
$\mathbf{F}_O^g$	Global object feature tensor, $n_O \times n_C \times d$
$\mathbf{F}_S^\ell$	Scene patch-level feature matrix, $H'W' \times d$
H', W'	Height/width of feature grid; $H'W'$ patches per image
$\mathbf{f}_S^\ell[p]$	Feature of patch p in scene S
$\mathbf{f}_+$	Object-like prototype in MPG
τ_+	Similarity threshold for foreground-like patches in MPG
α	Fraction of patches retained after MPG filtering

$\mathbf{m}_S$	Binary foreground mask over scene patches in MPG
G^i	Grounding triple $(\mathbf{Q}_O^i, \mathbf{K}_O^i, \mathbf{V}_O^i)$ at layer i
G_+^i	Aggregated positive grounding triple at layer i
$\mathbf{Q}_O^i, \mathbf{K}_O^i, \mathbf{V}_O^i$	Object queries, keys, and values at layer i
$\mathbf{Q}_S^i, \mathbf{K}_S^i, \mathbf{V}_S^i$	Scene queries, keys, and values at layer i
$\mathbf{Q}_{G_+}^i$	Aggregated object query prototype at layer i
$\mathbf{m}_{\text{fg}}^i$	Foreground mask at layer i in Self-Grounding
τ_G	Similarity threshold for foreground assignment in SG
$\mathbf{A}_+^i, \mathbf{A}_-^i$	Attention outputs for foreground and background groups at layer i
$\llbracket \mathbf{A}_+^i, \mathbf{A}_-^i \rrbracket$	Position-aware concatenation of group outputs
$\text{Atten}(\cdot)$	Standard multi-head self-attention operator
$\mathbf{M}_i$	i -th segmentation mask returned by SAM
q_i	IoU score (SAM stability/confidence) associated with mask $\mathbf{M}_i$
τ_I	IoU threshold used internally by SAM
δ	Mask threshold offset used to measure stability in CW (reuses symbol; context disambiguates)
s_i	Stability score of mask $\mathbf{M}_i$
τ_s	Stability cutoff used to retain reliable masks
CW	Set of IoU scores that pass the stability filter
SC_m	Final object matching score (semantic/appearance/geometry), before CW weighting

$[\text{cls}]$	Learnable classification token in the Transformer
$[\text{cls}]_{\text{emb}}$	Embedding of the $[\text{cls}]$ token at the input of a layer
$\mathbb{S}^{d-1}$	Unit sphere in $\mathbb{R}^d$

INTRODUCTION

Deep learning has become the standard tool for many computer vision tasks. Large neural networks, trained on big image or point cloud datasets, can reach very high accuracy in controlled settings. This progress is possible thanks to powerful hardware and the availability of large-scale data. However, real-world environments are often very different from the training conditions. As a result, models that work well in the lab can fail when they are deployed in practice, especially in industrial and robotic applications.

In this thesis, we focus on this gap between controlled benchmarks and real deployments. We are interested in models that must work on *3D point clouds* and on *model-based unseen objects*, such as industrial parts described by CAD models but never seen during training. These two settings arise naturally in industrial and robotic perception pipelines, where a system must operate continuously on streams of sensor data and support downstream tasks such as recognition, manipulation, and pose estimation under strict latency and reliability constraints. In these settings, collecting new labeled data and retraining the models is usually too slow, too expensive, or simply not possible. For point clouds in particular, acquisition conditions vary across depth sensors and production lines, and the resulting changes in noise, sparsity, and occlusion patterns can cause abrupt performance drops if the model is not robust to such shifts. For model-based unseen objects, the challenge is that new parts can appear after deployment, while only lightweight object specifications (e.g., CAD assets or a small number of reference renderings) are available, making per-object training and data collection impractical. We therefore look for methods that can adjust models at test-time, with little or no extra training, and with strong constraints on memory, speed, and data access.

Domain shifts in practice. The input data at inference time may differ from the training data in many ways: new sensors, new noise patterns, different backgrounds, lighting, or object materials. In 3D perception, these shifts are often structural: changes in depth sensing modality,

calibration, and reconstruction pipelines affect missing returns and depth noise; changes in distance and viewpoint alter sampling density and quantization; and occlusion and clutter produce partial observations that distort both local neighborhoods and global shape statistics. For point clouds, this can mean new scanners, different resolutions, or new types of clutter. Such effects are particularly important for transformer-based point cloud models, where tokenization and attention depend on the geometry and statistics of the observed point set; changes in sparsity and noise can therefore propagate through the representation in a systematic way. For industrial images, this can mean different cameras, factory lines, or object instances. When domain shift is present, the usual assumption that training and test data come from the same distribution is violated. This can lead to a large drop in performance, even if the model was strong on the original dataset.

Limits of retraining and fine-tuning. A simple idea to address domain shift is to retrain or fine-tune the model on new data that better matches the deployment environment. However, this is often not realistic. First, each new scenario could require a separate retraining process, which is expensive and hard to maintain. Second, in many cases, the model only sees data in an *online* or *episodic* way, where each sample or batch is observed once and then discarded. Third, retraining can be blocked by strict time or memory constraints, especially on embedded devices.

Lack of labels and source data. Even if retraining were computationally possible, it also needs labeled data. In many real-world systems, we do not have labels when we deploy the model. For industrial parts, labeling can require expert knowledge and careful measurement, which is slow and costly. In addition, the original training data (the *source* data) may not be available anymore, for privacy, storage, or business reasons. For example, in medical applications or in private industrial pipelines, it might be forbidden to store or reuse the original dataset. As a result, we need adaptation methods that work without labels and without access to the source data.

Test-Time Adaptation without backpropagation. Test-Time Adaptation (TTA) aims to adjust a pretrained model during inference, using only the incoming test samples, often without labels and without the source data. Many existing TTA methods still rely on backpropagation and gradient-based updates during deployment. While effective in some cases, this can be slow and memory-hungry, and it may be hard to integrate in real systems. This difficulty is amplified in 3D pipelines, where inference itself is already computationally demanding and where test samples often arrive as a stream under real-time or near-real-time constraints. Gradient-based test-time optimization typically requires multiple forward/backward passes, careful tuning of hyperparameters (e.g., learning rates, update frequency, and stopping criteria), and sometimes buffering of recent target samples—all of which can introduce latency, instability, and operational complexity in deployment. In this thesis, we explore a different direction: *backpropagation-free* and *training-free* adaptation. Instead of running optimization at test-time, we design simple and analytical updates that modify only specific parts of the model, such as normalization layers or token sets, in a single forward pass. This design choice aligns with the practical requirement of preserving predictable runtime and stable behavior while still exploiting inference-time signals for self-alignment under distribution shift.

Foundation models and unseen industrial objects. Recent work has produced powerful vision backbones and segmentation models trained on large and diverse datasets. Examples include Vision Transformers for point clouds and image encoders combined with segmentation models like SAM. These models can be reused across many tasks, but they are not perfect out of the box for industrial scenarios. In particular, *model-based unseen object* (MISUO) settings are very challenging. Here, we must detect and segment objects that are defined by CAD models or a few reference images, but that never appear in the training data. In practical manipulation and inspection pipelines, this segmentation step is often a prerequisite for reliable downstream processing (e.g., instance-level reasoning and pose estimation), which makes both accuracy and runtime critical. The setting is challenging because scenes may contain heavy clutter, partial

occlusions, and distractors with similar appearance or geometry; consequently, naive prompting or direct matching can be ambiguous, and segmentation hypotheses can drift to incorrect regions when the object is only partially visible. Existing methods often use text-based grounding or heavy proposal generation, which can be unreliable or too slow in practice. This motivates inference-time procedures that remain adaptation-free with respect to per-object training, but still incorporate structured self-correction signals, for example through prompt refinement and self-grounding mechanisms that consolidate and verify candidate hypotheses using model-based cues.

In this thesis, we study how to steer such models at inference time, using only simple operations on normalization, tokens, or visual prototypes, and without retraining. We aim to improve robustness under domain shift for 3D point clouds and to enable efficient and accurate instance segmentation of model-based unseen objects.

0.1 Research statement

The main goal of this thesis is to design and analyze *test-time methods* that make modern vision models more robust and more practical in real-world settings, especially in 3D and industrial applications. We focus on the following properties:

- The methods should work with *frozen backbones*, without retraining the whole network.
- The adaptation (or steering) should be *backpropagation-free* and as light as possible, often using a single forward pass.
- The methods should not require access to the original training data and should not rely on labels at deployment.
- For industrial unseen objects, the methods should use information that is easy to obtain, such as CAD models or a small set of reference images, instead of large annotated datasets.

To this end, we introduce three complementary approaches: (i) a careful analysis and reparameterization of Layer Normalization for point cloud test-time adaptation, (ii) a token-level purging mechanism for backpropagation-free adaptation in point cloud transformers, and (iii) a prototype-guided self-grounded prompting scheme that reuses large segmentation models for model-based unseen object segmentation without any training.

0.2 Contributions

The main contributions of this thesis are organized into three chapters, each based on a peer-reviewed or submitted research paper.

- **Chapter 2:** we focused on token-level adaptation in point cloud Transformers. We introduce *Purge-Gate (PG)*, a backpropagation-free TTA method that improves robustness by removing tokens that are strongly affected by domain shift before they reach the attention layers. PG computes a shift score for each token and purges the most shifted ones, allowing attention to focus on reliable regions. We present two variants: PG-SP, which uses source statistics when they are available, and PG-SF, which is fully source-free and uses a CLS-based prototype. Both variants work without gradients at test-time and bring large gains in accuracy, speed, and memory usage on multiple corrupted 3D point cloud datasets.

Related publication:

- Purge-Gate: Backpropagation-Free Test-Time Adaptation for Point Clouds Classification via Token Purging, submitted to the *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025. **Authorship:** First author. **Roles:** Method design, implementation, experiments, and writing.
- **Chapter 3:** we moved from token-level adaptation to revisit Layer Normalization (LN) from a domain perspective for point cloud models and show why many BatchNorm-style TTA tricks fail on Transformer backbones. We prove that, as the feature dimension and batch

size grow, the pre-affine LN outputs concentrate around zero mean and unit variance, so re-normalizing across batches is not necessary and can even be harmful. Based on this insight, we propose *LN-TTA*, a simple test-time adaptation method that keeps the pre-affine LN fixed and only adjusts the post-affine mean and scale in a single forward pass. LN-TTA is backpropagation-free, source-free, and very efficient in time and memory, while giving strong improvements on several corrupted 3D point cloud benchmarks.

Related publication:

- Revisiting Layer Normalization for Point Cloud Test Time Adaptation, accepted in the *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2026.
Authorship: First author. **Roles:** Method design, implementation, experiments, and writing.

- **Chapter 4:** we extend our focus from classification on point clouds to model-based unseen object instance segmentation in images. We propose *PRISMA*, a training-free framework that refines prompts for large segmentation models using visual prototypes derived from CAD models or multi-view reference images. PRISMA contains three key modules: (i) a Mask Proposal Generation (MPG) module that uses DINO-like embeddings and prototypes to propose object-like regions to segment, (ii) a Self-Grounding (SG) module that injects object prototypes into the backbone and separates foreground and background tokens through block-diagonal attention without retraining, and (iii) a Confidence-Weighted (CW) scoring scheme that uses the model’s own confidence signals to rank predictions. On the BOP benchmark, PRISMA improves instance segmentation quality and speed compared to previous MISUO methods, while remaining fully training-free.

Related publication:

- PRISMA: Prompt Refinement for Instance Segmentation of Model-based Unseen Objects, submitted to *Pattern Recognition*, 2025. **Authorship:** First author. **Roles:** Method design, implementation, experiments, and writing.

0.2.1 Additional contributions

Along with the main contributions, I made additional contributions to related 3D projects during this thesis, which provided me with a valuable understanding necessary for my main work.

- **Geomask3d:** Geometrically informed mask selection for self-supervised point cloud learning in 3d, published in the *Transactions on Machine Learning Research (TMLR)*, 2024. **Authorship:** Second author. **Roles:** Contributed to the method enhancement, validation, as well as writing the paper.
- **SMART-PC:** Skeletal Model Adaptation for Robust Test-Time Training in Point Clouds, published in the *International Conference on Machine Learning (ICML)*, 2025. **Authorship:** Second author. **Roles:** Contributed to the method inception, validation, as well as writing the paper.
- **Test-time adaptation in point clouds:** Leveraging sampling variation with weight averaging, published at *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2025. **Authorship:** Second author. **Roles:** Contributed to the method inception, implementation, and validation, as well as writing the paper.

0.2.2 External collaboration

This doctoral project was accompanied by an external collaborations:

- **Research chair ÉTS - Zebra Technologies (2025 - 2025)**, where we conducted applied research for industrial computer vision, mainly focusing on 3D perception and 6DOF pose estimation.

CHAPTER 1

LITERATURE REVIEW

1.1 Introduction

Real-world deployment of vision systems routinely violates the i.i.d. assumption: sensors drift, environments change, and devices differ. In industrial and robotic perception, this mismatch is not an occasional anomaly but an expected operating condition: point clouds may be acquired with different depth sensors and reconstruction pipelines, and objects may appear under new viewpoints, occlusions, clutter, and varying sampling densities. Moreover, many deployments must process a test stream under strict latency and memory constraints, which limits the feasibility of repeated optimization or per-environment re-training. This has long motivated the broader field of *Domain Adaptation* (DA), which studies how to transfer a model trained on a labeled *source* domain $\mathcal{D}_S = \{(x_i^S, y_i)\}_{i=1}^{N_S}$ to a different *target* domain $\mathcal{D}_T = \{x_j^T\}_{j=1}^{N_T}$ with mismatched distributions $P_S(x) \neq P_T(x)$. Classical *unsupervised* DA approaches reduce source–target discrepancies in feature space using adversarial alignment or statistic-matching and normalization strategies (Ganin *et al.*, 2016; Li, Wang, Shi, Liu & Hou, 2016; Iwasawa & Matsuo, 2021; Schneider *et al.*, 2020). In parallel, *Domain Generalization* (DG) aims to learn representations that are robust *before* deployment, e.g., by seeking flat minima and weight averaging, so that the model generalizes to unseen domains without any target data during training (Cha *et al.*, 2021; Izmailov, Podoprikin, Garipov, Vetrov & Wilson, 2018). A practical relaxation, *Source-Free Domain Adaptation* (SFDA), removes the need to store or access source data at deployment, adapting only with the frozen source model f_θ and unlabeled target samples (Liang, Hu & Feng, 2020; Yazdanpanah & Moradi, 2022). These settings align naturally with privacy-sensitive or storage-limited deployments, where models may be shipped without retaining the original training set.

Despite their impact, DA/DG methods typically assume an *offline* or *pre-deployment* adaptation stage with sufficient compute, repeated passes over target data, or access to auxiliary buffers, assumptions that can be incompatible with streaming, privacy-sensitive, or embedded settings.

This limitation is especially visible in 3D perception pipelines: the target distribution can change across facilities, sensors, and production batches, and perception must run continuously rather than as a one-time batch adaptation step. These constraints naturally lead to *Test-Time Adaptation* (TTA) as a focused subproblem within DA: the model f_θ must *self-adjust online at inference* using only the unlabeled test stream $\{x_j^T\}$, without revisiting source data or performing expensive retraining. Early TTA formulations minimized predictive uncertainty or leveraged augmentation consistency while restricting updates to small parameter subsets (Wang, Shelhamer, Liu, Olshausen & Darrell, 2020; Boudiaf, Mueller, Ben Ayed & Bertinetto, 2022; Zhang, Levine & Finn, 2022b); more recent work pushes toward *backpropagation-free* adaptation and 3D-specific designs (Mirza *et al.*, 2023; Wang *et al.*, 2024c). The appeal of these directions is primarily operational: gradient-based test-time optimization often requires multiple forward/backward passes, careful tuning of learning rates and stopping criteria, and sometimes buffering of recent samples, which can introduce latency and instability in real deployments—particularly for transformer-based point-cloud backbones where inference is already non-trivial.

Within TTA, research has diversified along four complementary axes: (i) *entropy-based* self-supervision (Wang *et al.*, 2020), (ii) *feature-level normalization* and distribution recalibration (Li *et al.*, 2016; Mirza, Micorek, Possegger & Bischof, 2022), (iii) *token/patch-level structure* in modern transformers for 2D/3D perception (Vaswani, 2017; Pang *et al.*, 2022; Yu *et al.*, 2022), and (iv) *gradient-free* mechanisms that avoid backpropagation entirely for stability and latency (Wang *et al.*, 2024c; Gao *et al.*, 2023). Although these axes are often presented abstractly, they are tightly connected to the constraints of practical perception systems: for example, token-level operations and normalization statistics can be modified at inference time with predictable compute, making them attractive in streaming 3D settings. A detailed comparison of DA, DG, and TTA is provided in Table 1.1.

Positioned within this DA→TTA trajectory, this thesis contributes three components: (1) **Purge-Gate (PG)** (see Chapter 2), a backpropagation-free 3D TTA framework based on entropy-guided token purging; (2) **LN-TTA** (see Chapter 3), a theoretical revisiting of Layer

Normalization as a closed-form, self-contained adaptation mechanism; and (3) **PRISMA** (see Chapter 4), an adaptation-free framework for prompt refinement and instance segmentation of model-based unseen objects. The third component connects the same deployment mindset to a complementary generalization regime: in model-based unseen object (MISUO) perception, the system must segment novel industrial parts specified by CAD models or reference renderings, where per-object training is infeasible and naive prompt-based segmentation can be brittle under clutter, occlusion, and distractors (Chen *et al.*, 2024b; Nguyen, Groueix, Ponimatin, Lepetit & Hodan, 2023; Hodan *et al.*, 2024). Together, these advances articulate a *forward-only* perspective on robustness, eschewing iterative optimization in favor of structural, statistical, and representational self-alignment at test time.

1.2 From Domain Shift to Test-Time Adaptation

1.2.1 Origins in Domain Adaptation

Early research into distribution shift mitigation focused on *Unsupervised Domain Adaptation* (UDA), wherein models trained on a labeled source domain $\mathcal{D}_S$ are adapted to an unlabeled target domain $\mathcal{D}_T$. Classical approaches minimized discrepancies between feature distributions across domains, using metrics such as Maximum Mean Discrepancy (MMD) (Arbel *et al.*, 2019), and adversarial alignment (Ganin *et al.*, 2016). While successful under controlled conditions, these methods relied on access to both source and target data during adaptation, making them unsuitable for deployment scenarios with privacy or storage constraints.

The introduction of *Source-Free Domain Adaptation* (SFDA) (Liang *et al.*, 2020) marked a major conceptual leap: adaptation could proceed using only the pre-trained source model f_θ and unlabeled target data. Yet, most SFDA methods still required iterative optimization or pseudo-labeling, approaches that remain computationally costly and susceptible to error accumulation. TTA can be viewed as an extreme limit of SFDA, where adaptation occurs online, per batch, with minimal compute.

Table 1.1 Comparison of *Domain Adaptation* (DA), *Domain Generalization* (DG), and *Test-Time Adaptation* (TTA) along practical, statistical, and deployment axes. Examples reference canonical methods from the literature review (see citations)

Aspect	DA	DG	TTA
Primary goal	Transfer a source-trained model to a <i>known</i> target distribution (often unlabeled) via adaptation.	Learn source-domain representations that <i>generalize</i> to unseen targets <i>without</i> any adaptation.	Adapt a frozen source model <i>online at inference</i> using only the test stream.
When adaptation happens	<i>Pre-deployment</i> / <i>offline</i> (or batch-wise before deployment).	<i>Training-time only</i> ; no target-side updates.	<i>During deployment</i> (online), per batch or per sample.
Data access (train/adapt)	Requires source data; typically requires access to target data (unlabeled) during the adaptation phase.	Source domains only (often multiple); no target data.	No source data; unlabeled target samples at test time (streaming).
Label requirements	Target labels: usually no; source labels: yes.	Only source labels; no target labels.	No labels at test time; uses unsupervised signals (entropy/consistency/statistics).
Typical supervision signal	Discrepancy minimization, adversarial alignment, statistic matching (e.g., BN/AdaBN, MMD, DANN) (Ganin <i>et al.</i> , 2016; Li <i>et al.</i> , 2016; Arbel, Korba, Salim & Gretton, 2019).	Regularization across sources (flat minima, ensembling/averaging) (Cha <i>et al.</i> , 2021; Izmailov <i>et al.</i> , 2018).	Entropy minimization, augmentation consistency, forward-statistics, gradient-free structure (Wang <i>et al.</i> , 2020; Zhang <i>et al.</i> , 2022b; Wang <i>et al.</i> , 2024c).
Optimization at deployment	Yes (adaptation stage).	No.	<i>Often no full backprop</i> (e.g., BFTT3D, PG, LN-TTA); some methods do light backprop (e.g., TENT).
Compute & memory budget at deployment	Moderate to high (offline adaptation passes).	Low (no target-time compute).	Low to moderate; can be <i>very low</i> with forward-only methods (PG, BFTT3D, LN-TTA).
Streaming / continual shift	Limited; re-adaptation needed when the target drifts.	Moderate; robustness depends on training diversity, not on-the-fly correction.	Strong; designed for <i>on-the-fly</i> correction under non-stationary streams.
Privacy / compliance	Often requires sharing/holding target samples during adaptation.	No target data needed.	No source data; uses only temporary test samples.
Robustness lever (where it acts)	<i>Feature-space</i> alignment (source $\leftrightarrow$ target) and classifier calibration.	<i>Parameter-space</i> flatness/diversity for broad invariance.	<i>Prediction/structure/statistics at test time</i> : entropy/consistency (TENT/MEMO), forward perturbation (BFTT3D), token purification (PG), analytic LN scaling (LN-TTA).
Works well when	A stable, accessible target set is available pre-deployment.	Many diverse sources approximate future targets; domain gaps are not extreme.	Targets arrive on-the-fly; source is unavailable; latency/compute are constrained.
Examples	DANN, MMD-based DA, AdaBN (Ganin <i>et al.</i> , 2016; Arbel <i>et al.</i> , 2019; Li <i>et al.</i> , 2016).	SWAD, weight averaging (Cha <i>et al.</i> , 2021; Izmailov <i>et al.</i> , 2018).	TENT, MEMO, BFTT3D, MATE, PG, LN-TTA (Wang <i>et al.</i> , 2020; Zhang <i>et al.</i> , 2022b; Wang <i>et al.</i> , 2024c; Mirza <i>et al.</i> , 2023; Yazdanpanah <i>et al.</i> , 2025).

1.2.2 The Emergence of Test-Time Training and Adaptation

Test-Time Training (TTT). Sun et al. introduced *Test-Time Training with self-supervision* (Sun et al., 2020), where a lightweight self-supervised head is attached during source training and then optimized *at test time* using only unlabeled target samples Figure 1.1. Concretely, TTT adapts a subset $\theta_{\text{ttt}} \subset \theta$ by minimizing a self-supervised objective on the incoming stream:

$$\min_{\theta_{\text{ttt}}} \mathbb{E}_{x_j^T \sim \mathcal{D}_T} \mathcal{L}_{\text{ss}}(h_{\theta_{\text{ttt}}}(f_{\theta}(\tau(x_j^T))), y_{\text{ss}}(\tau)), \quad (1.1)$$

where f_{θ} is the frozen backbone, $h_{\theta_{\text{ttt}}}$ the auxiliary head updated at test time, τ a test-time augmentation or masking operator, and $y_{\text{ss}}(\tau)$ a self-supervised target (e.g., rotation ID, jigsaw index, or masked-patch reconstruction target). Unlike TTA, which typically optimizes a *task-side* uncertainty (e.g., entropy) with minimal or no backprop, TTT optimizes a *self-supervised* loss at test time, trading extra backpropagation and latency for stronger invariance under shift. Variants (e.g., MAE-based) follow the same principle while changing $\mathcal{L}_{\text{ss}}$ and τ .

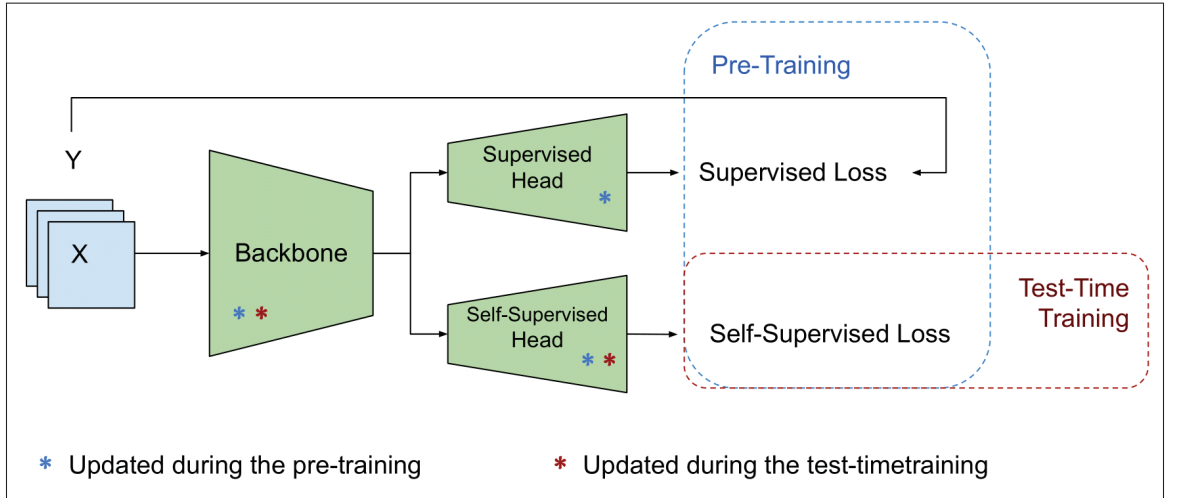


Figure 1.1 The overall schematic of TTT methods. During the pretraining, the backbone and its dual heads get updated. During the Test-Time Training, just the main backbone and self-supervised head get updated

Test-Time Adaptation (TTA). The foundational work of TENT (Wang *et al.*, 2020) introduced the principle of minimizing prediction entropy over unlabeled test batches:

$$\min_{\theta} \mathbb{E}_{x_j^T \sim \mathcal{D}_T} \mathcal{H}(p_{\theta}(y|x_j^T)), \quad (1.2)$$

where $\mathcal{H}(\cdot)$ denotes entropy and θ are the model parameters to be adapted. TENT restricted adaptation to affine parameters of Batch Normalization (BN) layers, avoiding full model retraining. Subsequent variants such as MEMO (Zhang *et al.*, 2022b) and parameter-free online TTA (Boudiaf *et al.*, 2022) improved robustness and efficiency, while backpropagation-free methods for 3D, such as BFTT (Wang *et al.*, 2024c) and MAE-based methods like MATE (Mirza *et al.*, 2023), extended these ideas to point cloud modality Figure 1.2.

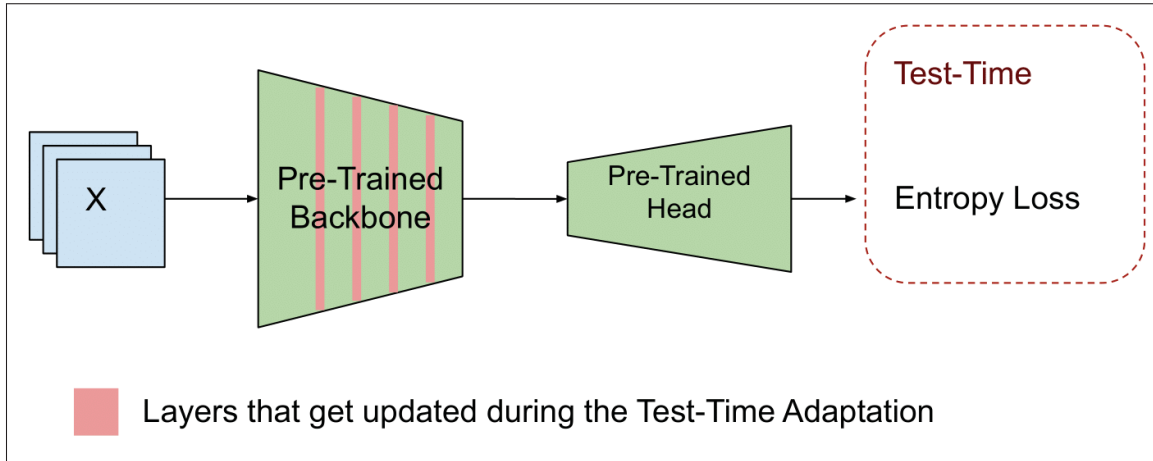


Figure 1.2 In Test-Time Adaptation, given a pretrained network with the task head, we only update a small subset of the network, using a label-agnostic objective

Despite these advances, BN-based TTA suffers from several limitations: it depends heavily on large batch sizes, lacks theoretical guarantees, and cannot be trivially applied to architectures like Transformers that rely on Layer Normalization. These challenges motivated new directions based on feature recalibration, token structure, and normalization theory, directions that this thesis systematically develops through PG, LN-TTA, and PRISMA.

1.3 Architectural Foundations for Adaptation

Convolutional networks excel on grid-structured signals, but their fixed local receptive fields and architectural coupling to spatial grids limit flexibility when inputs or objectives vary. Transformers, with token-based representations and content-adaptive self-attention, provide a uniform interface across modalities and tasks, supporting long-range dependencies and late fusion without redesigning inductive biases (Vaswani, 2017). Just as crucial for test-time methods, modern transformer stacks standardize on *Layer Normalization* (LN), which is stable under small-batch and streaming inference (Xiong *et al.*, 2020; Lyu, Li & Arora, 2022). These properties make transformers a natural substrate for our contributions: token-level selection (PG), analytic, backpropagation-free normalization updates (LN-TTA), and prompt refinement (PRISMA) that operate directly on tokens and LN-affine parameters rather than requiring heavyweight retraining.

1.3.1 Normalization as a Vehicle for Adaptation

Normalization layers play a pivotal role in TTA because they modulate distributional statistics at inference time. BN (Ioffe & Szegedy, 2015) leverages batch-wise mean and variance, while LN (Ba, Kiros & Hinton, 2016) operates per sample across features, making it inherently more stable under small-batch or online conditions. Studies such as (Lubana, Dick & Tanaka, 2021; Lyu *et al.*, 2022; Xiong *et al.*, 2020) have shown that normalization interacts deeply with optimization and representation geometry. LN in particular has been linked to implicit geometric effects and information decoupling across tokens, positioning it as a natural interface for adaptation without backpropagation.

However, LN’s theoretical properties under distributional shift remained underexplored until recent analytical work in *our* LN-TTA chapter (see Chapter 2) revealed its token-wise invariance and asymptotic behavior. This understanding forms the backbone of LN-TTA, which reformulates LN’s affine parameters (γ, β) as domain-calibration variables amenable to closed-form update rules rather than gradient descent.

1.3.2 Object-centric point cloud classification

A key premise in this thesis is that the 3D input is an *object-centric point set*. This differs from (i) standard image classification, where the signal is represented on a dense grid and background context can be informative, and from (ii) indoor/outdoor *scene-level* point cloud problems (e.g., RGB-D scans or LiDAR sweeps), where the input contains multiple semantic entities and large-scale background structure. In object-centric point cloud classification, each sample typically represents a single object instance (or a tight crop around it), and therefore contains little to no background geometry. The dominant sources of variability are instead tied to how the object surface is observed: partial visibility due to viewpoint and self-occlusion, missing returns and depth noise induced by sensing modality and material properties, and changes in sampling density arising from range, reconstruction settings, or preprocessing. This distinction is important for interpreting robustness and adaptation results in this thesis: the distribution shifts of interest primarily perturb the *geometry and sampling statistics of the object surface*, rather than background composition or scene layout.

The modern deep-learning treatment of point clouds is often traced to PointNet (Qi, Su, Mo & Guibas, 2017a), which explicitly models a point cloud as an *unordered set of vectors* and proposes a network that consumes raw point sets without voxelization or multi-view rendering. PointNet motivates direct point-set processing by noting that voxel grids incur substantial volumetric overhead and quantization artifacts, while multi-view pipelines introduce intermediate representations that may lose geometric fidelity. The PointNet formulation emphasizes three geometric requirements: permutation invariance to input ordering, meaningful geometric interactions rooted in the underlying distance metric, and robustness under rigid transformations. Formally, an object point cloud can be written as $X = \{p_i\}_{i=1}^N$ with $p_i \in \mathbb{R}^3$ (optionally augmented with additional channels such as normals). A point-set classifier should be invariant to permutations of $\{p_i\}$ and stable to common perturbations of sampling and sensing.

PointNet (Qi *et al.*, 2017a) achieves permutation invariance by applying a shared pointwise function $h(\cdot)$ to each point and aggregating the resulting features with a symmetric function

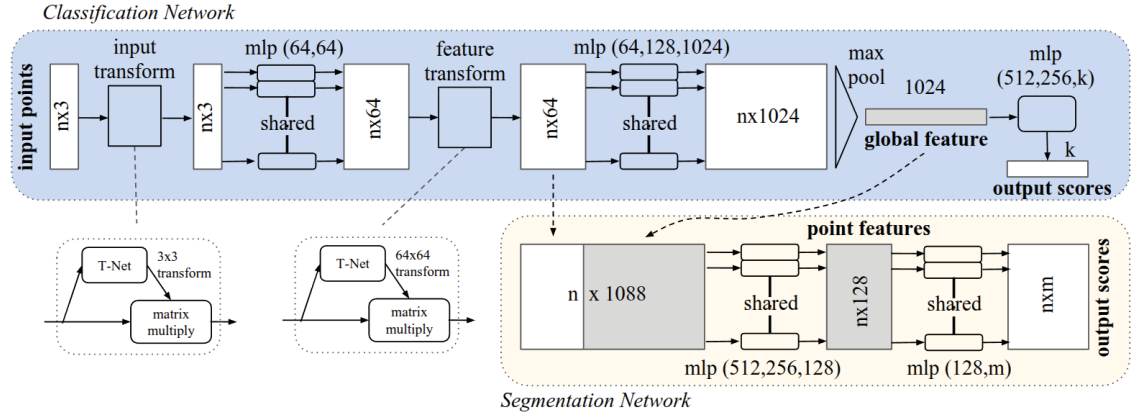


Figure 1.3 Canonical point-set learning pipeline (PointNet-style): shared pointwise embedding, symmetric aggregation (max pooling) to form a global descriptor, optional alignment modules (T-Net), and prediction heads for classification and per-point segmentation

$g(\cdot)$, instantiated by max pooling. This corresponds to the generic set-function form $f(\{x_i\}) \approx g(h(x_1), \dots, h(x_N))$, producing a global descriptor that is fed to an MLP head for classification Figure 1.3. PointNet further introduces an alignment module (T-Net) that predicts an affine transform for input points and a higher-dimensional feature transform, improving robustness to rigid motions while preserving the point-set structure. The paper also provides an instructive intuition for stability under common corruptions: the pooled descriptor is determined by a subset of points that achieve maximal activations (“critical points”), helping explain why point-set encoders can remain functional under point deletion (missing data) and point insertion (outliers). (Citation note: please add a BibTeX entry for PointNet. If you want one sentence on architectural evolution, you may also cite a small set of representative follow-ups, such as PointNet++ / DGCNN / KPConv / Point Transformer.)

1.3.3 Transformers and Tokenization in 3D Vision

The transformer architecture (Vaswani, 2017) introduced token-level processing as a general interface for visual data. However, the token representation and embedding differ based on the data modality. Compared to 2D images, 3D point clouds are a set of unordered and unorganized points in the 3D space represented using the points’ location (x, y, z) Figure 1.4. In 3D vision,

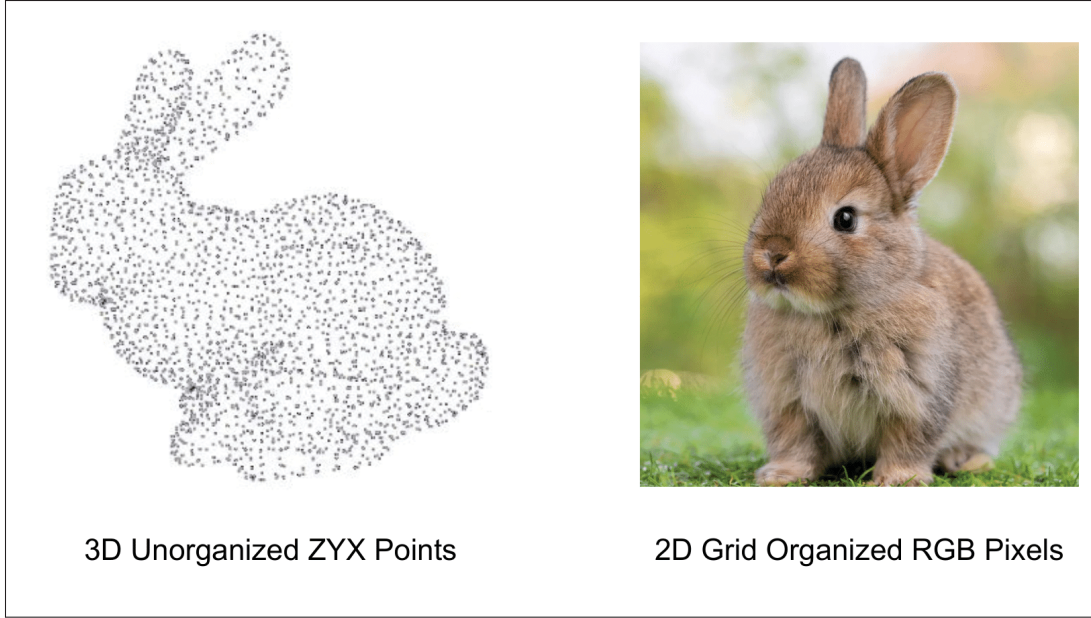


Figure 1.4 Compared to a 2D image, a point cloud is an unorganized and unordered set of 3D points of arbitrary size

an unordered point set $X_i = \{p_{i,n}\}_{n=1}^{N_p} \subset \mathbb{R}^3$ is turned into a sequence of *geometric tokens* (See Figure 1.5). Given an input cloud, Farthest Point Sampling (FPS) selects token centers $\{c_t\}_{t=1}^N$, and k-Nearest Neighbors gathers local neighborhoods $\mathcal{N}_t = \text{KNN}_k(c_t; X_i)$. Each token is recentered to its FPS center and embedded by a lightweight map $\mathcal{E}$ into $x_{i,t} \in \mathbb{R}^d$, producing the matrix $x_i \in \mathbb{R}^{N \times d}$ fed to the Transformer.

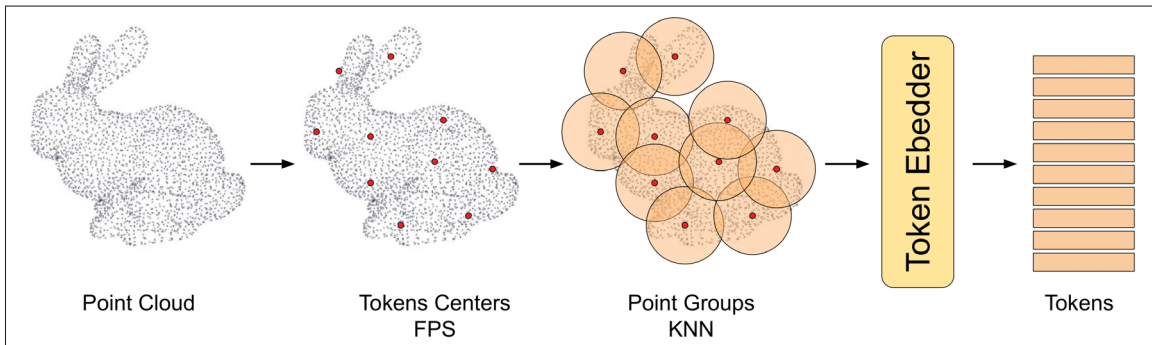


Figure 1.5 The process of tokenization of 3D point clouds into embedded tokens. These tokens are fed into Transformer layer for further processing

1.4 Test-Time Adaptation in 3D Perception

1.4.1 Challenges of Test-Time Shift in Point Clouds

Three-dimensional perception tasks, such as point-cloud classification and segmentation, amplify the brittleness of standard neural networks under distribution shift. Unlike 2D images, point clouds are unordered, non-Euclidean, and often exhibit variable density, occlusion, or sensor noise (Qi *et al.*, 2017a; Wang *et al.*, 2019b). *Compared to images on a regular pixel grid with locally consistent statistics (texture, color, illumination), point clouds are irregular samples of 3D geometry whose support can change from frame to frame.* As a result, common image corruptions (e.g., blur or brightness shifts) map to qualitatively different 3D distortions: missing points from self-occlusion, uneven sampling density with distance, quantization from depth discretization, and large pose/rotation variation. Moreover, many point sets lack texture or color entirely; thus, invariances learned from appearance cues in 2D do not transfer, and small geometric perturbations can propagate non-locally through attention on tokens/patches. Consequently, the latent representations learned by pre-trained models are highly sensitive to distributional distortions in geometry and sampling (Sun *et al.*, 2022).

The emergence of large-scale benchmarks such as ModelNet-C, ShapeNet-C, and ScanObjectNN-C (Sun *et al.*, 2022; Uy, Pham, Hua, Nguyen & Yeung, 2019; Chang *et al.*, 2015) exposed this fragility by simulating test-time corruptions spanning density shifts, Gaussian noise, background clutter, and rotational perturbations. Across these settings, networks trained on clean data frequently lose over 30–50% accuracy when evaluated under corrupted conditions.

While traditional robustness training approaches (e.g., adversarial augmentation or corruption-based fine-tuning) can partially alleviate this, they violate the deployment constraint of *no source access at inference*. Thus, TTA for 3D perception requires methods that can restore or recalibrate latent features purely from unlabeled, streaming test data, a problem compounded by the lack of Batch Normalization in most transformer-based architectures.

1.4.2 From BN-Based 2D TTA to Token-Level Adaptation in 3D Transformers

Early successes in 2D TTA, e.g., entropy minimization and augmentation consistency, operate through *Batch Normalization (BN)* by updating affine parameters or moments to reduce predictive uncertainty (Wang *et al.*, 2020; Zhang *et al.*, 2022b; Boudiaf *et al.*, 2022). In modern 3D pipelines, however, point-cloud backbones are Transformer-based and rely on *Layer Normalization (LN)*, which normalizes *per token across features* rather than across the batch (Pang *et al.*, 2022; Yu *et al.*, 2022). This architectural shift removes the cross-sample coupling exploited by BN-centric TTA, limiting the direct transfer of 2D recipes to 3D.

1.4.3 Baseline details

MATE (Mirza *et al.*, 2023). MATE (*Masked Test-Time Entropy*) adapts 3D point-cloud transformers by coupling an entropy-minimization objective with a masked-autoencoding (MAE) pretext at test time. A light reconstruction head is optimized online to stabilize predictions under corruption, while the task head benefits indirectly from the self-supervised signal. This design improves robustness over purely entropy-driven updates, but it still performs gradient-based optimization at inference and primarily injects a *global* adaptation signal rather than identifying and correcting token-local corruptions.

SMART-PC (Bahri *et al.*, 2025a). SMART-PC structures test-time training around corruption-invariant geometric cues (e.g., skeleton-like proxies) to regularize adaptation in 3D classification. By aligning self-supervised objectives with stable shape structure, it reduces overfitting to noisy artifacts and improves stability compared to naively optimizing under shift. While effective, SMART-PC still relies on backpropagation and does not explicitly target the removal of *locally* corrupted tokens that can dominate attention in transformer backbones.

TTT-Rot (Sun *et al.*, 2020). Test-Time Training with rotation prediction attaches a lightweight self-supervised head during source training and optimizes it at test time to induce invariance. The rotation (or similar pretext) objective provides label-free supervision on the incoming stream,

improving generalization under shift. However, the need for online optimization increases latency/memory, and the signal remains indirect for token-local defects typical of 3D corruptions.

SHOT (Liang *et al.*, 2020). SHOT transfers the *source hypothesis* (classifier) and adapts in a source-free manner by minimizing conditional entropy while enforcing class diversity to avoid collapse. It uses target-only clustering of features to refine decision boundaries without accessing source data. Although successful in 2D and some 3D settings, SHOT’s objective is largely *global* and was originally developed for BN-centric pipelines, offering limited leverage for LN-dominant 3D transformers.

T3A (Iwasawa & Matsuo, 2021). T3A is a *backpropagation-free* classifier-adjustment method: it forms class-specific prototypes from confident test samples and updates the final linear head on-the-fly (no gradients through the backbone). This yields fast, stable gains, but the adjustment acts only at the classifier, leaving internal representations—and token-level corruptions—unaddressed.

TENT (Wang *et al.*, 2020). TENT minimizes prediction entropy at test time while *only* updating BatchNorm affine parameters and/or moments, avoiding full retraining. Its simplicity and label-free nature made it a strong baseline in 2D. Yet TENT hinges on BN’s batch coupling; in 3D transformers that primarily use LN (token-wise normalization), TENT’s mechanism weakens and may exhibit strong batch-size sensitivity.

DUA (Mirza *et al.*, 2022). DUA (*The Norm Must Go On*) pursues dynamic, target-aware normalization by recalibrating statistics to better match test distributions, often improving stability without heavy architectural changes. While compelling for BN-driven networks, DUA’s moment-updating intuition is harder to port to LN-dominant transformers, where per-token normalization removes the explicit domain (batch) axis needed for AdaBN-style realignment.

BFTT3D (Wang *et al.*, 2024c). BFTT3D introduces a fully *backpropagation-free* 3D TTA framework with two streams: a frozen source model and a target stream built from a non-parametric encoder and subspace alignment to static class prototypes; the final prediction fuses

source/target logits via entropy-aware mixing. By avoiding pseudo-labeling and gradients, it is stable and efficient, but it depends on stored source prototypes and applies coarse, feature-level control rather than explicit token purification.

1.4.4 The Shift Toward Gradient-Free Adaptation

Gradient-based TTA, though effective, incurs latency, instability, and memory overhead due to backpropagation through large networks. Recent works have sought to remove gradients entirely from the adaptation loop. Examples include diffusion/feature-perturbation strategies (Gao *et al.*, 2023), forward-only adaptation for 3D (Wang *et al.*, 2024c). This paradigm shift is not merely computational; it redefines what it means for a model to “adapt”: no longer optimizing parameters, but reconfiguring its own representation space through self-consistent updates.

Token pruning vs. robustness. Efficiency-focused *token pruning* removes low-utility tokens (see Figure 1.6) to cut FLOPs (e.g., learned pruning, dynamic pruning for segmentation, pruning&pooling in ViTs) (Kim *et al.*, 2022; Tang, Zhang, Liu, Liu & Liu, 2023; Wu *et al.*, 2023). These techniques assume stable token distributions and are not shift- or uncertainty-aware; they do not distinguish redundant tokens from tokens corrupted by domain shift, and therefore cannot *self-correct* under unseen conditions.

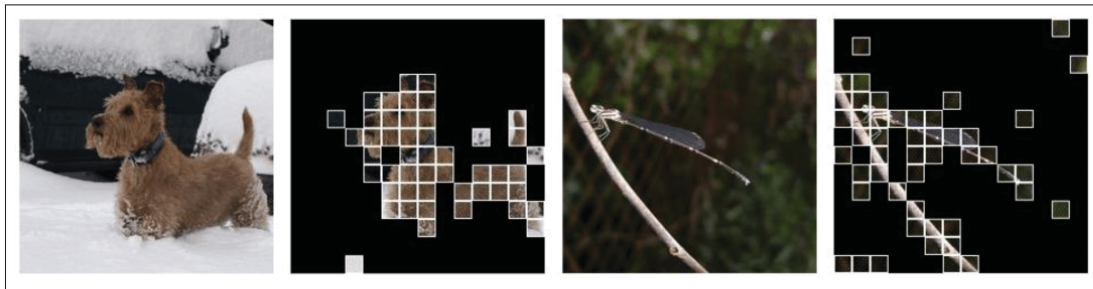


Figure 1.6 In token pruning, the goal is to remove the background and unimportant tokens to reduce the memory and computation footprint of transformer networks.

Adapted from (Wu *et al.*, 2023)

Motivation for token-level purification. Under 3D shifts, a small subset of corrupted or out-of-distribution tokens can dominate attention and propagate errors, degrading global representations.

This motivates *token-level* remedies that act where corruption arises. Our **Purge-Gate (PG)** (see Chapter 2) reframes pruning from a compression heuristic into an adaptive, robustness mechanism: instead of removing tokens for speed, PG *purifies* the token set by discarding those with high entropy or statistical inconsistency during a single forward pass. In doing so, PG bridges efficiency-style token selection with TTA’s uncertainty signals, providing localized, backpropagation-free correction that complements feature-level normalization and addresses the core limitation of prior, predominantly global, 3D TTA strategies.

1.4.5 Normalization-Based Adaptation

Our **Purge-Gate (PG)** (see Chapter 2) was designed to tackle *structural* failures first: under 3D domain shifts (sensor noise, occlusion, irregular sampling), a small subset of corrupted tokens can dominate attention and derail the representation. PG *purifies* the token set by removing high-entropy or statistically inconsistent tokens in a single forward pass, delivering robust gains without backpropagation. Yet, ablations revealed a residual gap: even after structural purification, features could remain *statistically miscalibrated*, their per-token scales and offsets drifted between source and target. This observation pointed to normalization as a complementary lever: if we could re-center and re-scale activations *analytically* at test time, PG’s structural gains might be stabilized and amplified.

Normalization layers are central to distributional robustness, but most early test-time strategies implicitly assumed *Batch Normalization (BN)* as the adaptation interface. In BN-centric models, the *domain dimension* is the mini-batch itself, so updating running moments or affine parameters naturally realigns the model to the test distribution. In Transformer backbones, however, *Layer Normalization (LN)* is predominant and computes statistics *per token across features*, not across samples in a batch (Ba *et al.*, 2016; Xiong *et al.*, 2020; Lyu *et al.*, 2022). This architectural fact removes cross-sample coupling and deprives us of a direct “domain axis” on which to recalibrate moments at test time.

1.4.5.1 BN-based adaptation and its limitation under architectures with LN.

A representative BN-based approach is **AdaBN** (Li *et al.*, 2016), which simply replaces source BN statistics with target-domain estimates at inference, often yielding surprisingly strong gains with no labels and minimal overhead. AdaBN’s effectiveness stems from BN’s batch-level (i.e., domain-level) normalization: moments aggregated over test samples act as a proxy for the target domain and can be swapped in place of source moments. With LN, this mechanism breaks: statistics are computed *within each token* and recomputed every forward pass, so there is no persistent, batch-aggregated notion of “target-domain moments” to adopt. Consequently, AdaBN-style test-time realignment is *not directly applicable* to LN-based Transformers Figure 1.7.

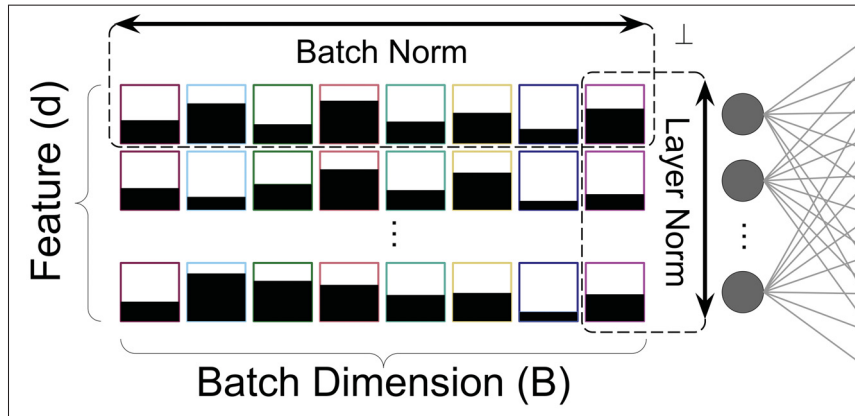


Figure 1.7 Comparison of BatchNorm and LayerNorm as adaptation interfaces. BatchNorm couples samples through batch moments, enabling moment swapping at test time. LayerNorm normalizes within each token across features, with no batch coupling, so there is no persistent domain axis to realign

1.4.5.2 Why an LN-native solution is needed.

Because LN does not normalize along the domain (batch) dimension, BN-era fixes (AdaBN-style moment swapping) cannot be ported. What we require is an *LN-native*, backpropagation-free update that (i) respects LN’s per-token formulation, (ii) leverages only forward-pass test statistics, and (iii) yields closed-form, stable corrections to counter domain-induced mean/variance drift.

1.4.5.3 Our answer: LN-TTA.

Motivated by the above, we develop **LN-TTA** (see Chapter 3), which reinterprets LN as a *domain-alignment operator* and derives *closed-form* recalibrations of its affine parameters from test-time moments, without gradients, labels, or batch coupling. LN-TTA thus complements PG: PG enforces *structural* robustness by purifying tokens; LN-TTA enforces *statistical* robustness by analytically re-centering and re-scaling per-token features. Together, they form a forward-only pathway from token-level purification to principled normalization alignment in Transformer-based 3D perception.

1.5 Adaptation-Free Segmentation and Prompt Refinement: The PRISMA Framework

Promptable foundation models (e.g., SAM) have reshaped 2D segmentation by enabling zero-shot masks from prompts (Kirillov *et al.*, 2023; Liu *et al.*, 2023). Classical two-stage detectors (Faster/Mask R-CNN) are reliable when labels/semantics are known (Girshick, 2015; He, Gkioxari, Dollár & Girshick, 2017), and category-level generalization methods (ShapeMask, NOCS) inject priors to handle unseen instances within broad classes (Kuo, Angelova, Malik & Lin, 2019; Wang *et al.*, 2019a). However, they remain tied to category semantics and struggle with custom or textureless objects common in industry.

The *model-based* setting formalized by BOP (Hodan *et al.*, 2024) moves beyond categories: given a CAD or small multi-view set, systems must segment previously unseen, object-specific instances *without training* (MISUO). In this task, the algorithm needs to detect all the instances of each object of interest in a scene and then individually segment them, given a model of the object. (as known as model-based instance segmentation of unseen objects) Figure 1.8-a. Recent pipelines embrace SAM as a universal proposer but differ in how they select relevant masks. ZeroPose (Chen *et al.*, 2024b) densely queries SAM across the *entire* scene, then matches mask embeddings to the reference object; CNOS (Nguyen *et al.*, 2023) adopts the same generate-then-filter pattern and strengthens selection with appearance/semantic consistency against object templates; SAM6D (Lin *et al.*, 2024) further fuses appearance, semantic, and

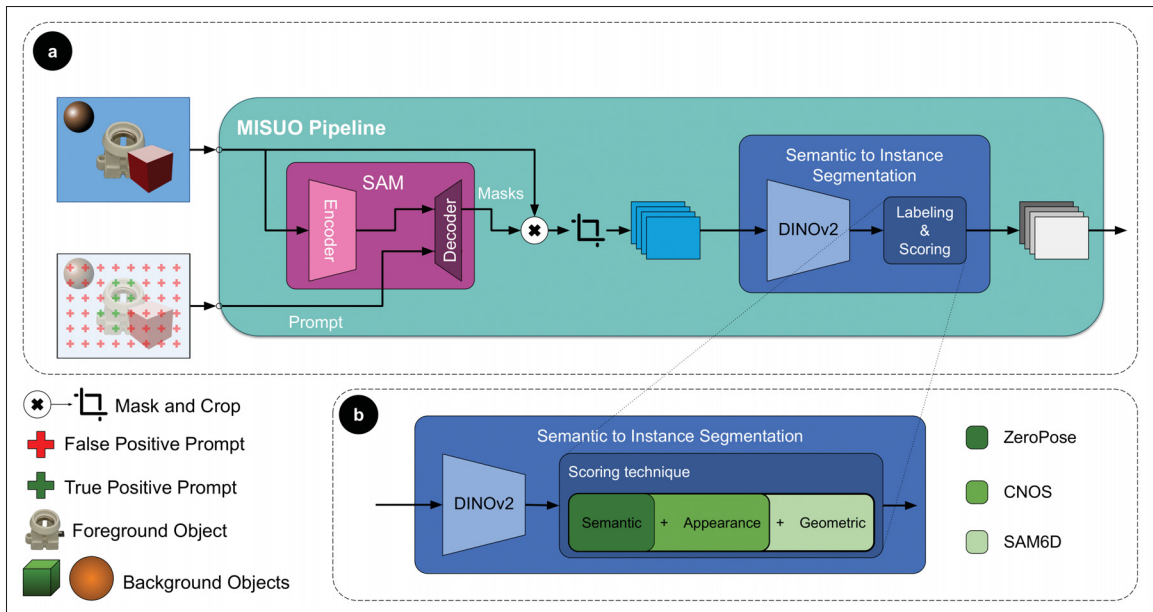


Figure 1.8 The MISUO, pipeline from ZeroPose to SAM6D. **(a)** In this pipeline, the object’s model is utilized to detect the instance-based segmentation of all instances of objects of interest (foreground objects). **(b)**, Former methods (ZeroPose, CNOS, SAM6D) have focused on improving the scoring techniques, neglecting the effect of over-prompting on the algorithm’s efficiency

geometric cues to refine candidates. Across these systems, the Achilles’ heel is the same: **full-scene prompting** (See Figure 1.8-b). Because SAM is queried everywhere and background proposals are discarded post hoc, computation and latency scale with image size rather than object likelihood, an unfeasible cost for embedded or real-time robotics.

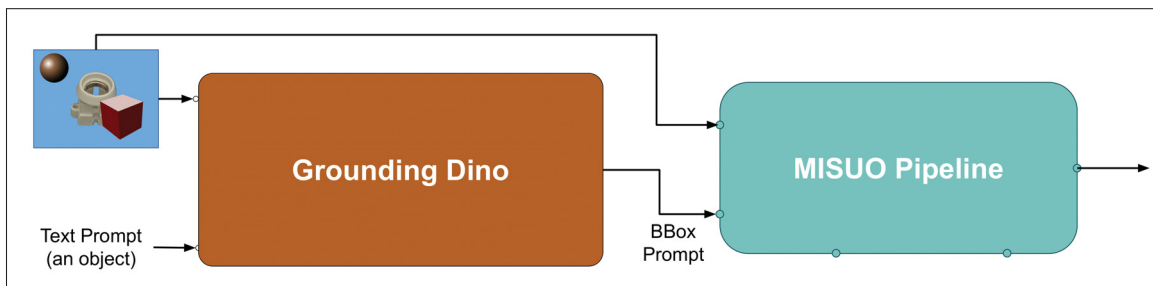


Figure 1.9 Grounded-SAM utilizes Grounding-DINO, and a text prompt to find the instance Bounding Boxes (BBox). In industrial settings, due to the MMP issue, objects can not clearly be defined using text prompts

A recent line reduces proposals via *text* prompts. Grounding-DINO localizes boxes from language using cross-modal grounding, which SAM then segments (Liu *et al.*, 2023); Grounded-SAM integrates the two for text-driven segmentation (Ren *et al.*, 2024). While this trims queries, it is brittle in industrial contexts, the MMP, because many parts are visually near-identical yet linguistically indistinct (e.g., T-LESS objects) and thus remain indistinguishable by text alone.

1.5.1 Baselines details

ZeroPose. **ZeroPose** tackles MISUO with a training-free, CAD-driven *generate-then-select* strategy for instance segmentation (Chen *et al.*, 2024b). It first *discovers* candidates by densely prompting SAM across the *entire* scene to produce a large pool of class-agnostic masks $\{\mathbf{M}_i\}$. Each candidate mask is then *associated* with the object model by comparing its visual embedding (extracted from the masked image crop) to embeddings of the CAD rendered under diverse synthetic views, and ranked by similarity with geometric/visibility consistency checks to suppress distractors. The final instance masks are obtained by keeping high-scoring candidates with non-maximum suppression and simple refinement. This training-free design scales across unseen objects, but its reliance on exhaustive full-scene prompting and large candidate pools incurs notable computational cost and increases the chance of background proposals passing early filters.

CNOS. **CNOS** targets *model-based* instance segmentation directly by utilizing model cues: it first generates a large pool of class-agnostic proposals with SAM (or FastSAM), then filters them by comparing proposal features against *rendered templates* of the CAD model using a vision backbone (e.g., DINOv2) to score semantic/appearance similarity (Nguyen *et al.*, 2023). Additional geometric/semantic checks remove inconsistent masks, which improves robustness for textureless or category-ambiguous parts; however, the method still relies on exhaustive proposal generation and template matching, so runtime scales with the initial SAM proposal count and template bank size.

SAM6D. **SAM6D** addresses MISUO with a training-free, model-based *proposal–scoring* pipeline built on SAM (Lin *et al.*, 2024). The *Instance Segmentation Model (ISM)* densely queries SAM over the *entire* scene to obtain class-agnostic masks, then assigns each mask an *object matching score* by fusing *appearance* (image features from a strong visual encoder on the masked crop), *semantics* (label priors tied to the object identity when available), and *geometry* (CAD-consistency cues from multi-view render features). Masks are ranked and filtered (e.g., via NMS and thresholding) to yield the final instance segmentations, improving precision over single-cue selection and suppressing distractors in clutter. As with ZeroPose/CNOS, however, segmentation latency is dominated by **full-scene prompting**: SAM is queried everywhere and background proposals are discarded post hoc, so compute scales with image size rather than object likelihood.

Grounding-DINO. **Grounding-DINO** reduces proposals by conditioning detection on *text* prompts: cross-modal grounding aligns language tokens with image regions so that a phrase produces localized boxes that can be forwarded to a segmenter (Liu *et al.*, 2023). In MISUO this trims queries compared to full-scene prompting, but performance hinges on the availability of discriminative textual descriptors—an issue in industrial settings where visually near-isomorphic parts lack reliable language cues (the MMP).

Grounded-SAM (NIDS-Net). **Grounded-SAM** composes Grounding-DINO with SAM: text-conditioned boxes from Grounding-DINO are converted into instance masks by SAM, yielding a text-to-segmentation pipeline with fewer SAM queries than full-scene prompting (Ren *et al.*, 2024), Lu, Guo, Ruozi, Xiang *et al.* (2024). While attractive for interactive scenarios, its dependence on language means the MMP persists on datasets like T-LESS, where objects are linguistically indistinct; in such cases, mask recall/precision degrade despite reduced proposal counts.

Despite the current progresses in the field, yet in industrial and robotic settings, deployment faces two persistent obstacles: (i) the computational burden of *exhaustive* full-scene prompting

when no external cue is available, and (ii) the *Modality Mismatch Problem (MMP)*, where text-based grounding fails to disambiguate visually similar, often textureless parts.

1.5.1.1 What is missing.

From these trajectories, two gaps are clear:

1. **Efficiency gap:** Proposal generation remains *exhaustive* (ZeroPose, CNOS, SAM6D), wasting queries on background.
2. **Grounding gap:** Proposal reduction via *text* (Grounding-DINO, Grounded-SAM) inherits cross-modal fragility (MMP) and fails on near-isomorphic industrial parts.

To address these, we introduce **PRISMA** (*Prompt Refinement for Instance Segmentation of Model-based unseen objects with Adaptation-free Self-Grounding*; see Chapter 4).

1.5.1.2 Positioning within the thesis’ forward-only paradigm.

PRISMA complements our 3D contributions by extending the same philosophy, *robustness without backpropagation*, to 2D segmentation. Where Purge-Gate removes corrupted tokens structurally, and LN-TTA corrects statistics analytically, PRISMA (i) *sparsifies* prompts with a model-based prototype to avoid wasteful full-scene querying (answering the efficiency gap highlighted by ZeroPose/CNOS/SAM6D), and (ii) *grounds* purely in the visual space to bypass the MMP that limits Grounding-DINO/Grounded-SAM. In doing so, it delivers a practical, deployment-ready MISUO pipeline that preserves the precision of model-based approaches while achieving the efficiency typically associated with prompt-driven systems, *without* any training or language dependence.

1.6 Benchmarking and Evaluation Protocols

This section introduces standard benchmarks, metrics, and evaluation protocols used by prior work on (i) Test-Time (Training/Adaptation) robustness for point-cloud **classification**, and (ii) **Model-Based Instance Segmentation of Unseen Objects (MISUO)**.

1.6.1 Test-Time Point-Cloud Classification

1.6.1.1 Benchmarks and corruption taxonomies.

Three corrupted dataset suites are widely adopted to assess robustness under distribution shift: **ModelNet40-C** (Sun *et al.*, 2022), **ShapeNet-C** (Chang *et al.*, 2015; Mirza *et al.*, 2023), and **ScanObjectNN-C** (Uy *et al.*, 2019; Mirza *et al.*, 2023). Each injects a standardized set of 15 test-time corruptions drawn from three families, *geometric transforms*, *additive noise*, and *density variations*, to mimic realistic acquisition artifacts such as background clutter, LiDAR sparsity, and sensor noise (Sun *et al.*, 2022). ShapeNet-C follows the open-source corruption pipeline of Sun *et al.* (2022) but applies it to the ShapeNetCore-v2 test split (Chang *et al.*, 2015; Mirza *et al.*, 2023). ScanObjectNN-C perturbs only the original ScanObjectNN test set with the same 15 object-level corruptions (Uy *et al.*, 2019; Mirza *et al.*, 2023).

1.6.1.2 Backbones and pretraining.

Prior work commonly evaluates on Transformer backbones pre-trained with masked point modeling, e.g., **PointMAE** (Pang *et al.*, 2022) as prepared by Mirza *et al.* (2023). Alternative backbones include Point-BERT (Yu *et al.*, 2022) and more recent state-space models such as **Point-Mamba** (Liang *et al.*, 2024), which probe architectural portability of test-time methods.

1.6.1.3 Problem setting and protocols.

The community distinguishes (i) *Test-Time Training (TTT)*, which optimizes a self-supervised auxiliary loss online (e.g., MAE reconstruction) using unlabeled test data (Sun *et al.*, 2020; Mirza *et al.*, 2023; Bahri *et al.*, 2025a), from (ii) *Test-Time Adaptation (TTA)*, which adapts using prediction-driven criteria such as entropy minimization (Wang *et al.*, 2020; Zhang *et al.*, 2022b; Boudiaf *et al.*, 2022) or backpropagation-free perturbations (Wang *et al.*, 2024c). Two evaluation protocols are standard: *episodic* (reset model/buffers after each test batch) and *online/streaming* (no reset). Because many modern 3D Transformers rely on **LayerNorm (LN)** rather than BN,

BN-centric updates may require care or resetting residual BN layers for fairness (Ba *et al.*, 2016; Xiong *et al.*, 2020; Lyu *et al.*, 2022; Wang *et al.*, 2020). Typical test-time batch sizes are small (e.g., $B=32$) to reflect deployment constraints; some methods report sensitivity to B .

1.6.1.4 Baselines by adaptation mechanism.

Representative **TTT** baselines include **TTT** with self-supervision (Sun *et al.*, 2020) and 3D-specific **MATE** (masked reconstruction + entropy) (Mirza *et al.*, 2023), as well as structured TTT like **SMART-PC** (Bahri *et al.*, 2025a). **TTA** baselines include **TENT** (BN-affine entropy minimization) (Wang *et al.*, 2020), **MEMO** (consistency under test-time augmentations) (Zhang *et al.*, 2022b), parameter-free classifier refinement **T3A** (Iwasawa & Matsuo, 2021), and **backpropagation-free** updates such as **BFTT3D** (Wang *et al.*, 2024c). Token-structural approaches like **Purge-Gate (PG)** target local corruption by removing high-entropy or statistically inconsistent tokens before attention (Yazdanpanah *et al.*, 2025).

1.6.1.5 Metrics and reporting.

The primary metric is **Top-1 accuracy (%)** per corruption and the *mean across all 15 corruptions*. It is customary to also report the *delta over Source Only*. Because TTA/TTT trade accuracy for latency/footprint, works report *runtime per batch*, *GPU memory*, and whether *gradients/optimizers* are used. For reproducibility, common practice fixes: (i) the exact corruption implementation of Sun *et al.* (2022); (ii) the same public pre-trained weights (e.g., PointMAE from Mirza *et al.* 2023); (iii) a single-pass adaptation budget (no multi-pass augmentation unless explicitly stated). Deviations (e.g., using different backbones or additional data) are highlighted to avoid confounds.

1.6.2 Model-Based Instance Segmentation of Unseen Objects (MISUO)

1.6.2.1 Benchmark: BOP and datasets.

MISUO is evaluated on the **BOP** benchmark (Hodan *et al.*, 2024), which focuses on *unseen, object-specific* instances provided via CAD or a few reference views, without per-object training. BOP aggregates diverse industrial and tabletop datasets: **LM-O** (Brachmann *et al.*, 2014), **T-LESS** (Hodan *et al.*, 2017), **ITODD** (Drost, Ulrich, Bergmann, Hartinger & Steger, 2017), **HB** (Kaskman *et al.*, 2019), **YCB-V** (Xiang, Schmidt, Narayanan & Fox, 2017), **TUD-L** and **IC-BIN** (Hodan *et al.*, 2024), covering textureless parts, clutter, occlusion, and lighting variability.

1.6.2.2 Tasks and evaluation metrics.

BOP supports (i) **instance segmentation** and **detection** and (ii) downstream **6DoF pose estimation**. For segmentation/detection, BOP follows COCO-style AP: per-object AP (APO) aggregates into per-dataset AP (APD) and a global **Mean** across datasets (Lin *et al.*, 2014). Evaluation considers up to 100 predictions/image and masks with at least 10% visible area. For pose, BOP reports **Average Recall (AR)** under VSD, MSSD, and MSPD; the dataset score (ARD) averages these and the **Mean** averages across datasets (Hodan *et al.*, 2024). Runtime is a first-class metric due to industrial constraints; many works report *time per instance/scene*.

1.6.2.3 Method families and inputs.

Training-free MISUO pipelines commonly leverage **SAM** for segmentation proposals and **DINOv2** features for matching (Kirillov *et al.*, 2023; Oquab *et al.*, 2023). **ZeroPose** densely queries SAM on the full scene and retains masks whose embeddings match the reference (Chen *et al.*, 2024b). **CNOS** filters SAM proposals using category geometry/semantics and 3D templates (Nguyen *et al.*, 2023). **SAM6D** integrates SAM with multi-cue similarity (appearance/semantics/geometry) for robust mask selection (Lin *et al.*, 2024). These approaches

are *train-free* but differ in prompt budgets and efficiency: full-scene prompting is accurate yet expensive. Language-grounded detectors (e.g., Grounding-DINO) reduce proposal counts but face the *modality mismatch problem* (MMP) for near-indistinguishable industrial parts (Liu *et al.*, 2023; Ren *et al.*, 2024).

1.6.2.4 Reporting conventions and fairness.

Works specify (i) the segmentation backbone (SAM vs. FastSAM), (ii) the visual backbone for matching (e.g., DINOv2), (iii) whether *extra backbones* (e.g., CLIP/DINO) are used, and (iv) any *test-time adaptation* of backbones (e.g., NIDS variants) when contrasting with strictly *adaptation-free* pipelines (Radford *et al.*, 2021; Zhang *et al.*, 2022a; Hodan *et al.*, 2024). Because MISUO is often a stage in broader pipelines, studies also report (v) *time per instance/scene* and (vi) downstream *pose AR* to quantify end-to-end impact (Wen *et al.*, 2024; Örnek *et al.*, 2025). To avoid hardware confounds, several works *re-run baselines* on the same machine and disclose thresholds (e.g., similarity cut-offs) to match prompt budgets across methods.

1.7 Concluding Remarks

This chapter situates our work within the trajectory from DA/DG to TTT/TTA, emphasizing deployment constraints, the architectural shift from BN to LN, and the need to act directly on token structure and per-token statistics rather than relying on batch coupling. The review motivates a forward-only perspective on robustness, favoring structural, statistical, and representation-aware corrections over iterative re-learning, and frames our contributions (**PG**, **LN-TTA**, **PRISMA**) as complementary steps toward that goal. Looking ahead, the key challenge is to consolidate accuracy–latency–memory trade-offs under small-batch, streaming conditions while preserving theoretical grounding and practical simplicity. In sum, adaptation is recast as the restoration of self-consistency within the model’s elements, a view developed in the chapters that follow.

CHAPTER 2

PURGE-GATE: BACKPROPAGATION-FREE TEST-TIME ADAPTATION FOR POINT CLOUDS CLASSIFICATION VIA TOKEN PURGING

Moslem Yazdanpanah* , Ali Bahri , Mehrdad Noori , Sahar Dastani , Gustavo Adolfo Vargas Hakim , David Osowiechi , Ismail Ben Ayed , Christian Desrosiers

Department of Software Engineering and IT, École de Technologie Supérieure,
1100 Notre-Dame West, Montreal, Quebec, Canada H3C 1K3

Paper published in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2025

Abstract

Test-time adaptation TTA is crucial for mitigating performance degradation caused by distribution shifts in 3D point cloud classification. In this work, we introduce Token Purging (PG), a novel backpropagation-free approach that removes tokens highly affected by domain shifts before they reach attention layers. Unlike existing TTA methods, PG operates at the token level, ensuring robust adaptation without iterative updates. We propose two variants: PG-SP, which leverages source statistics, and PG-SF, a fully source-free version relying on CLS-token-driven adaptation. Extensive evaluations on ModelNet40-C, ShapeNet-C, and ScanObjectNN-C demonstrate that PG-SP achieves an average of +10.3% higher accuracy than state-of-the-art backpropagation-free methods, while PG-SF sets new benchmarks for source-free adaptation. Moreover, PG is 12.4× faster and 5.5× more memory efficient than our baseline, making it suitable for real-world deployment. Code is available at <https://github.com/MosyMosy/Purge-Gate>

2.1 Introduction

Deep neural networks have recently achieved remarkable success in classifying 3D point clouds (Qi *et al.*, 2017a; Qi, Yi, Su & Guibas, 2017b; Wang *et al.*, 2019b; Pang *et al.*, 2022; Zhang *et al.*, 2022c; Zhang, Wang, Qiao, Gao & Li, 2023b; Bahri *et al.*, 2024a). However, their performance can degrade significantly when the test distribution (*target*) differs from the

training distribution (*source*). In 3D data, such shifts can arise from various factors, including differences in sensor types and occlusions. Given the vast range of possible distribution shifts, pretraining a model for every scenario is impractical. Test-Time Adaptation (TTA) mitigates

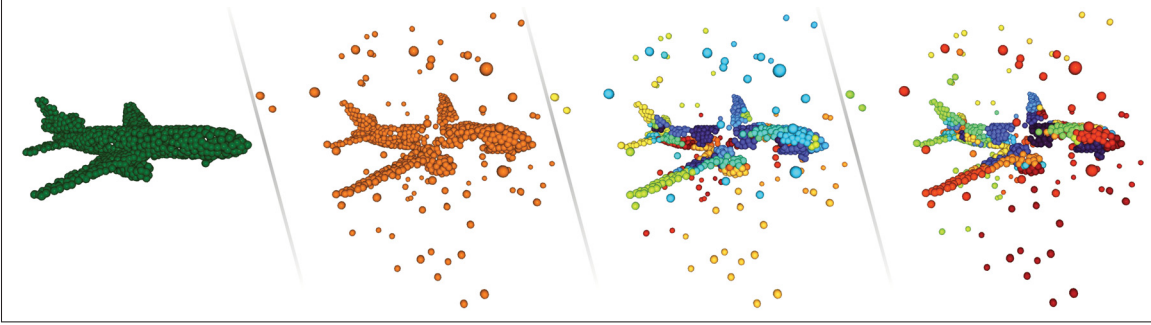


Figure 2.1 Token-level domain shift visualization. From left to right: clean point cloud, corrupted point cloud, tokens ranked by PG-SF, and tokens ranked by PG-SP. The color gradient from red to green indicates the degree of domain shift, with red representing the highest shift and green the lowest

distribution shifts that arise after training (Gandelsman, Sun, Chen & Efros, 2022; Gao *et al.*, 2023; Iwasawa & Matsuo, 2021; Lim, Kim, Choo & Choi, 2023; Yeo, Kar, Sodagar & Zamir, 2023; Schneider *et al.*, 2020) by enabling a source-pretrained model to adapt dynamically during inference using only unlabeled test data. Along with light weight transfer learning methods (Yazdanpanah *et al.*, 2022; Yazdanpanah & Moradi, 2022), various TTA methods have been proposed for 2D image analysis (Osowiechi *et al.*, 2024a,b; Hakim *et al.*, 2025; Noori *et al.*, 2025), including techniques that adjust batch statistics and affine parameters through entropy minimization (Nado *et al.*, 2020; Wang *et al.*, 2020), optimize a Laplacian-based clustering loss (Boudiaf *et al.*, 2022), or average the output distributions of multiple augmentations of individual test samples (Zhang *et al.*, 2022b). In contrast to TTA in the image modality, where we assume a fixed-dimensional input with a uniformly distributed target noise over the image area, test-time noise in point clouds can manifest in more complex ways. It may introduce new points, thereby expanding the dimensionality of the point cloud, or appear as unevenly distributed perturbations that affect the original structure. Consequently, applying similar adaptation strategies to point clouds presents unique challenges due to their non-Euclidean nature and inherent structural sparsity.

In recent years, numerous studies have focused on developing TTA methods for point cloud classification (Wang *et al.*, 2024c; Mirza *et al.*, 2023; Wang *et al.*, 2024d; Bahri *et al.*, 2024b). While these approaches have shown promising results, they come with notable limitations. For example, TENT (Wang *et al.*, 2020), as a dominant approach for image TTA, performs poorly when applied to point clouds. MATE (Mirza *et al.*, 2023) employs a Masked Autoencoder (MAE) reconstruction objective for both model pretraining and test-time adaptation. As a result, it falls under Test-Time Training approaches, which depend on auxiliary training-time tasks and cannot be applied to arbitrarily pre-trained models. Similarly, the method in (Bahri *et al.*, 2024b) adapts the model using multiple-point sub-sampling augmentations, significantly increasing computational cost. Finally, these methods iteratively adapt the model through multiple backpropagation steps. Consequently, they are unsuitable for runtime-critical applications and are sensitive to the number of update steps. BFTT3D (Wang *et al.*, 2024c) was recently proposed as a backpropagation-free method for TTA of point cloud data. Despite its adaptation-free advantages, it relies heavily on detailed class-level source prototypes, making it dependent on source data during the test time.

On the other hand, a recent line of research has focused on improving the efficiency of transformer (Vaswani, 2017) networks by optimizing the number of tokens, a technique known as Token Pruning. Introduced by (Kim *et al.*, 2022), these methods aim to identify and remove redundant or uninformative tokens. By eliminating these unnecessary tokens, they enhance the inference efficiency of transformers without altering the backbone parameters. Inspired by Token Pruning and motivated by the challenge posed by domain shift, we introduce an innovative application of token filtering to the task of TTA. Our approach focuses on detecting and removing tokens that are highly affected by domain shift at test time. Unlike Token Pruning, which targets unimportant tokens to improve computational efficiency, our method identifies tokens most impacted by domain shift Figure 2.1 and enhances TTA performance. *To emphasize its focus on filtering corrupted tokens, we refer to our approach as Token Purging.* Unlike existing methods, which have the above-mentioned limitations, our approach mitigates distribution shift at the token level. We observe that noise from domain shifts disrupts the attention mechanism in

transformer-based architectures, impairing feature aggregation. To counter this, we propose PG, a lightweight, backpropagation-free mechanism that detects and filters noisy tokens before they reach the attention layers. By dynamically refining the token set at inference time, our method improves performance under domain shifts without requiring additional optimization steps.

Our contributions can be summarized as follows:

- We formalize the effect of domain shift on the attention mechanism in point cloud Transformers. Based on this, we introduce Purge-Gate (PG), a backpropagation-free mechanism that selectively removes highly divergent tokens, thus improving test-time adaptation performance.
- We propose two variants of PG, Source-Prototype based Purge-Gate (PG-SP), which utilizes source statistics when available, and Source-Free Purge-Gate (PG-SF), operating in a fully source-free manner.
- We validate our approach across multiple benchmarks, including datasets with standard synthetic corruption and real-world datasets with domain shift (for the first time), using Transformer (Vaswani, 2017) and Mamba (Gu & Dao, 2023) (for the first time) architectures. We show that our method outperforms existing TTA methods while maintaining computational efficiency.

By addressing the challenges specific to point cloud adaptation, our method advances the field of test-time adaptation and offers a practical solution for improving model robustness in 3D Test-Time Adaptation (TTA).

2.2 Related Works

Test-Time Point Cloud Adaptation. Research on Test-Time Adaptation (TTA) for 3D point clouds remains limited compared to 2D image domains. One of the earliest methods in this area, MATE (Mirza *et al.*, 2023), employs a Masked Autoencoder (MAE) pretraining objective to enhance model robustness against distribution shifts. Following MATE, SMART-PC (Bahri *et al.*, 2025a), has proposed learning the object’s skeleton as an auxiliary task to update the network during training time. However, since MATE and SMART-PC rely on auxiliary training-time

tasks, they fall under Test-Time Training (TTT) rather than pure TTA. Similarly, MM-TTA (Shin *et al.*, 2022) integrates predictions from multiple modalities to reinforce self-supervised signals in 3D semantic segmentation. While effective in multimodal settings, its reliance on multiple input modalities makes it less applicable to single-modality point cloud classification.

BFTT3D (Wang *et al.*, 2024c) introduces a backpropagation-free TTA method specifically designed for 3D data. It mitigates domain shifts through a two-stream architecture that retains both source and target domain knowledge. However, it relies on class-specific prototypes derived from the source data, making it incompatible with fully source-free TTA settings.

Unlike these methods, our approach does not require auxiliary training objectives, multimodal data, or reliance on class-level source prototypes. Instead, we introduce a token-level adaptation mechanism that directly mitigates the effects of domain shift during inference in a backpropagation-free manner.

Token Pruning. Token pruning is a recent technique designed to improve the efficiency of transformer models by reducing the number of tokens during inference (Kim *et al.*, 2022). DToP (Tang *et al.*, 2023) enhances semantic segmentation efficiency by dynamically pruning tokens based on their difficulty, enabling early exits while preserving contextual information. The Pruning & Pooling Transformers (PPT) framework (Wu *et al.*, 2023) optimizes Vision Transformers (ViTs) by combining token pruning and pooling, thereby reducing computational cost without requiring additional trainable parameters. While these approaches primarily aim to improve inference speed by eliminating redundant tokens, they do not address the challenge of domain shifts. Our method takes a different direction: rather than pruning unimportant tokens for efficiency, we identify and remove tokens most affected by distribution shift, thereby enhancing TTA performance in a backpropagation-free manner.

Mamba Architecture. Mamba (Gu & Dao, 2023) is a variation of State Space Models that is introduced to the computer vision field by the VMamba paper (Liu *et al.*, 2024). Later, PointMamba (Liang *et al.*, 2024) adapted this architecture to point cloud modality. There are numerous works aimed at enhancing this architecture (Dastani *et al.*, 2025; Bahri *et al.*, 2025b),

but no empirical evaluation in a standard TTA setting. To show the versatility of our method, for the first time, we have used the Mamba network in a point cloud TTA setting.

2.3 Method

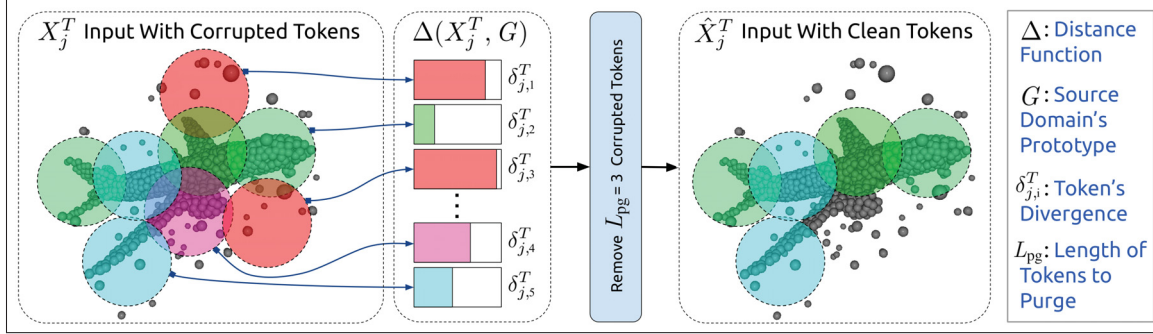


Figure 2.2 The overall procedure of the PG algorithm at the input of the attention layer

A point cloud is an unordered set of 3D points

$$X_i = \{p_{i,1}, p_{i,2}, \dots, p_{i,N_p}\} \subset \mathbb{R}^3, \quad (2.1)$$

where N_p is the number of points in sample i . Following (Pang *et al.*, 2022), each sample X_i is tokenized by selecting N token centers (typically $N \ll N_p$) via farthest point sampling (FPS), and forming local neighborhoods around them. We initialize $c_1 \in X_i$ (arbitrary) and, for $t = 2, \dots, N$,

$$c_t = \arg \max_{p \in X_i} \min_{c' \in \{c_1, \dots, c_{t-1}\}} \|p - c'\|_2, \quad (2.2)$$

$$\mathcal{N}_t = \{p \in X_i \mid p \in \text{KNN}_k(c_t; X_i)\}, \quad (2.3)$$

where t denotes the token index and $\mathcal{N}_t$ is the k -nearest-neighbor patch around c_t . A token embedding map $\mathcal{E}$ produces d -dimensional token vectors,

$$x_i = \begin{bmatrix} \mathcal{E}(c_1, \mathcal{N}_1)^\top \\ \vdots \\ \mathcal{E}(c_N, \mathcal{N}_N)^\top \end{bmatrix} \in \mathbb{R}^{N \times d}, \quad x_{i,t} \equiv x_i[t] \in \mathbb{R}^d. \quad (2.4)$$

A Transformer backbone processes x_i and a classifier head produces logits $\hat{y}_i = f_\theta(x_i)$.

We denote the labeled *source* training set and unlabeled *target* test set by

$$\mathcal{D}_S = \{(x_i^S, y_i)\}_{i=1}^{N_S}, \quad \mathcal{D}_T = \{x_j^T\}_{j=1}^{N_T}, \quad (2.5)$$

with $P_S(x) \neq P_T(x)$. During test-time adaptation (TTA) we seek an adapted model $f_{\theta'}$ that maximizes classification accuracy on P_T while keeping the backbone weights frozen and avoiding backpropagation.

A straightforward approach is to fine-tune θ on $\mathcal{D}_T$ using a label-free objective such as entropy minimization (e.g., TENT (Wang *et al.*, 2020)). However, under large domain shifts, incorrect pseudo-labels can be reinforced, leading to collapse. Instead, we focus on how test-time noise affects the self-attention mechanism and derive a *backpropagation-free* gating scheme—Purge-Gate (PG)—that filters corrupted tokens at the input of attention.

2.3.1 Attention Under Distribution Shift

In each self-attention layer, the interaction between model weights θ and token embeddings $x_{i,t}$ determines how information is aggregated. We examine how this interaction is perturbed by test-time noise and use this to motivate PG.

Let a noisy target token be

$$x_{j,t}^T = x_{j,t}^S + \epsilon_{j,t}, \quad (2.6)$$

where $x_{j,t}^S$ is the (unknown) clean embedding and $\epsilon_{j,t}$ is an unknown perturbation with magnitude $\|\epsilon_{j,t}\| = \|x_{j,t}^S - x_{j,t}^T\|$. Inside a Transformer layer, LayerNorm (LN) acts *per token* across features. For any token $x_t \in \mathbb{R}^d$ we follow the standard definition of LN:

$$\text{LN}(x_t) = \gamma \odot \frac{x_t - \mu_d(x_t)\mathbf{1}}{\sqrt{\sigma_d^2(x_t) + \varepsilon}} + \beta, \quad (2.7)$$

where $\gamma, \beta \in \mathbb{R}^d$ are learnable affine parameters, $\odot$ denotes elementwise multiplication, $\varepsilon > 0$ is the LN stability constant, $\mathbf{1} \in \mathbb{R}^d$ is the all-ones vector, and

$$\mu_d(x_t) = \frac{1}{d} \sum_{j=1}^d (x_t)_j, \quad \sigma_d^2(x_t) = \frac{1}{d} \sum_{j=1}^d ((x_t)_j - \mu_d(x_t))^2. \quad (2.8)$$

Proposition (Local Lipschitz Continuity of LN). *Let $x \in \mathbb{R}^d$ and define LN as in Equation (2.7). Assume we restrict attention to a compact set $\mathcal{D} \subset \mathbb{R}^d$ on which the per-token standard deviation satisfies $\sigma_d(x) \geq \sigma_{\min} > 0$ for all $x \in \mathcal{D}$. Then LN is locally Lipschitz on $\mathcal{D}$: there exists a finite constant $M > 0$ such that for any $x_{j,t}^S, x_{j,t}^T \in \mathcal{D}$,*

$$\|\text{LN}(x_{j,t}^S) - \text{LN}(x_{j,t}^T)\| \leq M \|\epsilon_{j,t}\|. \quad (2.9)$$

Sketch of Proof. Away from $\sigma_d(x) = 0$, LN is a composition of smooth maps: (1) $x \mapsto \mu_d(x)$ is linear; (2) $x \mapsto \sigma_d(x)$ is smooth where $\sigma_d(x) > 0$; (3) $x \mapsto 1/\sigma_d(x)$ is smooth on $(0, \infty)$; (4) the affine map $x \mapsto \gamma \odot x + \beta$ is smooth. Hence LN is continuously differentiable on any subregion that stays clear of $\sigma_d(x) = 0$. Because $\mathcal{D}$ is compact and contained in that region, the Jacobian $J_{\text{LN}}(x)$ is continuous and attains a finite maximum, say $M = \max_{x \in \mathcal{D}} \|J_{\text{LN}}(x)\|$. Applying the mean-value theorem to any $x_{j,t}^S, x_{j,t}^T \in \mathcal{D}$ yields

$$\|\text{LN}(x_{j,t}^S) - \text{LN}(x_{j,t}^T)\| \leq \|J_{\text{LN}}(\xi)\| \|\epsilon_{j,t}\| \leq M \|\epsilon_{j,t}\|, \quad (2.10)$$

for some ξ on the line segment between $x_{j,t}^S$ and $x_{j,t}^T$, confirming (2.9).

For $x \in \mathcal{D}$ with moderate noise, the noisy token $x_{j,t}^T$ remains close to $x_{j,t}^S$ after LN due to Equation (2.9). When noise dominates, i.e., $\|\epsilon_{j,t}\| \gg \|x_{j,t}^S\|$, the token behaves like a random vector and we have $\text{LN}(x_{j,t}^T) \approx \text{LN}(\epsilon_{j,t})$.

Proposition (Random Directions in High Dimension). *Let $u, v \in \mathbb{R}^{d'}$ be random vectors drawn from a sufficiently isotropic distribution on the sphere $\mathbb{S}^{d'-1}$. Then for large d' ,*

$$\mathbb{E}[u^\top v] \approx 0, \text{ and } \text{Var}(u^\top v) \approx \frac{1}{d'}. \quad (2.11)$$

Sketch of Proof. *When u, v are i.i.d. random points on $\mathbb{S}^{d'-1}$, the distribution of their dot product is sharply concentrated near zero, due to classical concentration-of-measure results: distinct random directions in high dimensions are nearly orthogonal. Hence, the mean and variance of $u^\top v$ scale as in Equation (2.11), with variance shrinking on the order of $1/d'$.*

Because LN enforces per-token zero mean and unit variance, noise-dominant embeddings after LN roughly lie on a sphere of radius $\sqrt{d}$. From Equation (2.11), two such random tokens $u, v \in \mathbb{R}^d$ satisfy $u^\top v \approx 0$.

In a self-attention layer, queries and keys are

$$Q_{j,t}^T = \text{LN}(x_{j,t}^T) W_Q, \quad K_{j,t'}^T = \text{LN}(x_{j,t'}^T) W_K, \quad (2.12)$$

with $W_Q, W_K \in \mathbb{R}^{d \times d}$ and $Q_{j,t}^T, K_{j,t'}^T \in \mathbb{R}^d$. The attention score between tokens t and t' is

$$S_{t,t'} = (Q_{j,t}^T)^\top K_{j,t'}^T. \quad (2.13)$$

If both tokens are noise-dominant, their directions are close to random and $S_{t,t'} \approx 0$ for $t' \neq t$. As noise magnitude increases, all scores concentrate near zero, and the softmax attention becomes nearly uniform:

$$A_{t,t'} = \frac{\exp(S_{t,t'}/\sqrt{d})}{\sum_{k=1}^N \exp(S_{t,k}/\sqrt{d})} \approx \frac{1}{N}, \quad (2.14)$$

leading to uninformative averaging rather than meaningful token interaction. Thus, from Equations (2.9) and (2.14), stronger domain shifts yield more noise-dominant tokens and hence more degraded attention.

2.3.2 Purge-Gate (PG)

Based on this analysis, we propose *Purge-Gate* (PG), a backpropagation-free gating mechanism that removes highly corrupted tokens before attention. Given a target sample $x_j^T \in \mathbb{R}^{N \times d}$, PG assigns each token a scalar *divergence score* via a distance function Δ from a source-domain prototype G :

$$\Delta(x_j^T, G) = (\delta_{j,1}^T, \delta_{j,2}^T, \dots, \delta_{j,N}^T). \quad (2.15)$$

Based on these token-level distances, PG removes the L_{pg} most divergent tokens from x_j^T , forming a *purified* token sequence

$$\hat{x}_j^T = x_j^T \setminus \{x_{j,t}^T \mid t \in \arg \max_{S \subseteq [N], |S|=L_{\text{pg}}} \sum_{t \in S} \delta_{j,t}^T\}, \quad (2.16)$$

where L_{pg} is the *purging size* and the remaining number of tokens is $\hat{N} = N - L_{\text{pg}}$. Keeping L_{pg} fixed preserves batch-wise parallel computation in the attention layers.

The main challenge is to construct a robust prototype G and a divergence measure Δ without assuming a known noise model. We therefore introduce two PG variants:

- **Source-Prototype Purge-Gate (PG-SP)**, which uses source-domain summary statistics as G ;
- **Source-Free Purge-Gate (PG-SF)**, which builds G from the learned [cls] token, without accessing source samples.

2.3.2.1 Source-Prototype Purge-Gate (PG-SP)

From Equation (2.9), the source-domain embeddings furnish a natural reference for detecting shifted tokens. We extract a prototype G from $\mathcal{D}_S$ in two ways.

a) Source embeddings via Welford’s algorithm. For each source sample $x_i^S \in \mathbb{R}^{N \times d}$ we define

$$\mu_i = \frac{1}{N} \sum_{t=1}^N \mathcal{E}(x_{i,t}^S), \quad (2.17)$$

$$\sigma_i = \sqrt{\frac{1}{N} \sum_{t=1}^N \left(\mathcal{E}(x_{i,t}^S) - \mu_i \right)^2}, \quad (2.18)$$

and accumulate global source statistics (μ_S, σ_S) using Welford’s online updates:

$$\phi_S \leftarrow \phi_S + \frac{1}{n} (\phi_i - \phi_S), \quad \phi \in \{\mu, \sigma\}, \quad (2.19)$$

where n is the running sample count.

b) Modified LN accumulation. Alternatively, we modify the first LayerNorm layer (defined in Equation (2.7)) so that, during source pretraining, it accumulates per-token means and variances via Equation (2.19). This yields the same (μ_S, σ_S) without storing raw source data.

In both cases, we take $G = (\mu_S, \sigma_S)$ with $\mu_S, \sigma_S \in \mathbb{R}^d$. We then define Δ as a Mahalanobis distance between target tokens $x_{j,t}^T$ and G :

$$\Delta(x_j^T, G) = (\delta_{j,1}^T, \dots, \delta_{j,N}^T), \quad \delta_{j,t}^T = \sqrt{(x_{j,t}^T - \mu_S)^\top \Sigma_S^{-1} (x_{j,t}^T - \mu_S)}, \quad (2.20)$$

where $\Sigma_S = \text{diag}(\sigma_S^2)$ is the diagonal covariance matrix. Tokens far from the source prototype under this metric are purged.

2.3.2.2 Source-Free Purge-Gate (PG-SF)

PG-SP requires either source data (variant *a*) or a model retrained with modified LN (variant *b*), which may be impractical. To avoid these constraints, we build a prototype G solely from the pre-trained model.

In a classification Transformer f_θ , the learnable [cls] token is optimized during pretraining to summarize the content of the other tokens. Given a source sequence

$$x_i^S = ([\text{cls}], x_{i,1}^S, \dots, x_{i,N}^S) \in \mathbb{R}^{(N+1) \times d}, \quad (2.21)$$

the self-attention layer forms

$$Q_i = \text{LN}(x_i^S)W_Q, \quad K_i = \text{LN}(x_i^S)W_K, \quad V_i = \text{LN}(x_i^S)W_V, \quad (2.22)$$

and updates [cls] as

$$[\text{cls}]' = \sum_{t=0}^N \alpha_{0t} V_{i,t}, \quad (2.23)$$

where

$$\alpha_{0t} = \frac{\exp(S_{0t}/\sqrt{d})}{\sum_{t'=0}^N \exp(S_{0t'}/\sqrt{d})}, \quad S_{0t} = Q_{i,0}^\top K_{i,t}. \quad (2.24)$$

Gradients flowing through this update force [cls] to attend to informative input features, so that its final representation approximates a prototype of the source token distribution (Zou, Yi, Li & Li, 2024b).

In PG-SF, we therefore define the prototype at test time as

$$G = \text{LN}([\text{cls}])W_Q \in \mathbb{R}^d, \quad (2.25)$$

and measure divergence in the query/key space using cosine distance. For the target sample x_j^T , we compute

$$\tilde{K}_j^T = \text{LN}(x_j^T)W_K \in \mathbb{R}^{N \times d}, \quad (2.26)$$

and set

$$\Delta(x_j^T, G) = (\delta_{j,1}^T, \dots, \delta_{j,N}^T), \quad \delta_{j,t}^T = -\frac{\tilde{K}_j^T[t, :] G}{\|\tilde{K}_j^T[t, :]\|_2 \|G\|_2}, \quad (2.27)$$

so that tokens less aligned with the prototype (larger $\delta_{j,t}^T$) are preferentially purged.

2.3.3 Backpropagation-Free Considerations

2.3.3.1 Test-Time Hyperparameter Selection

In PG, PG-SP, and PG-SF, the purging length L_{pg} is crucial, but cannot be learned by gradient-based optimization at test time. Moreover, its optimal value is corruption-dependent, making exhaustive search infeasible.

We therefore adopt a *parallel* test-time selection scheme. For each input x , we evaluate a small candidate set

$$\mathcal{L}_{\text{pg}} = \{L_1, L_2, \dots, L_K\}, \quad (2.28)$$

run the PG mechanism for each $L \in \mathcal{L}_{\text{pg}}$, and select the logits with minimum entropy:

$$L_{\text{pg}}^* = \arg \min_{L \in \mathcal{L}_{\text{pg}}} H(f(x; L)), \quad (2.29)$$

where $f(x; L)$ denotes the model logits for purging length L , and $H(\cdot)$ is the entropy:

$$H(z) = - \sum_c p_c \log p_c, \quad p_c = \frac{\exp(z_c)}{\sum_{c'} \exp(z_{c'})}. \quad (2.30)$$

This yields near-optimal L_{pg} with a modest number of forward passes and *no* backpropagation.

2.3.3.2 Normalization Statistics Update

In many TTA methods, the procedure comprises:

- **Adaptation:** update a subset of parameters θ on a target batch using a test-time loss;
- **Inference:** evaluate accuracy on the same batch.

During adaptation, BatchNorm (BN) statistics may update implicitly, in addition to explicit parameter updates (Ioffe & Szegedy, 2015).

Our method is backpropagation-free and has no explicit adaptation pass: token purging is applied only during inference. Consequently, BN moving averages trained on the source domain may no longer match the target statistics. To mitigate this, we reset BN statistics before each forward pass, forcing BN to use the current batch statistics rather than incompatible train-time estimates, in line with prior TTA practice (Ioffe & Szegedy, 2015). This keeps PG compatible with models that still contain BN layers while remaining stateless and backpropagation-free.

2.4 Experiments

This section provides a detailed evaluation of the proposed method across multiple 3D point cloud datasets. To assess its robustness and generalization capabilities, we perform experiments on three benchmark datasets: ModelNet40-C (Sun *et al.*, 2022), ShapeNet-C (Chang *et al.*, 2015), and ScanObjectNN-C (Uy *et al.*, 2019). These datasets incorporate a range of real-world challenges, including varying degrees of corruption and noise, allowing us to demonstrate the robustness of our approach across diverse and complex conditions. Additionally, we assess the performance of our method in standard adaptation settings, where the model is reset after each batch adaptation.

2.4.1 Implementation Details

For adaptation, we have used the publicly available pre-trained PointMAE (Pang *et al.*, 2022) model from MATE (Mirza *et al.*, 2023) for all the reproduced results to ensure a fair and consistent comparison. The adaptation process was carried out with a batch size of 32 and a single iteration in the standard mode. We have set $\mathcal{L}_{pg} = \{0, 2, 4, 8, 16, 32\}$ across all the experiments. Since our method does not involve backpropagation, it eliminates the need for an optimizer, making the adaptation process more efficient. To maintain consistency across all configurations, all experiments were conducted on a single NVIDIA A6000 GPU.

2.4.2 Datasets

ModelNet40-C. ModelNet40-C (Sun *et al.*, 2022) serves as a robustness evaluation dataset for point cloud classification, specifically designed to test the ability of architectures to handle distribution shifts encountered in real-world scenarios. It extends the original ModelNet40 test set by introducing 15 corruption types, grouped into three main categories: transformation, noise, and density variations. These corruptions replicate real-world challenges, such as sensor inaccuracies and LiDAR scan noise, offering a comprehensive framework for assessing model resilience under diverse perturbations.

ShapeNet-C. ShapeNetCore-v2 (Chang *et al.*, 2015) is a large-scale dataset widely used for point cloud classification, comprising 51,127 3D shapes spanning 55 categories. The dataset is structured into three subsets: 70% for training, 10% for validation, and 20% for testing. To evaluate model robustness in real-world scenarios, (Mirza *et al.*, 2023) introduced 15 corruption types to the test set, mirroring those found in ModelNet40-C. These corruptions were generated using an open-source implementation provided by (Sun *et al.*, 2022), resulting in the modified dataset known as ShapeNet-C.

ScanObjectNN-C. ScanObjectNN (Uy *et al.*, 2019) is a real-world point cloud classification dataset comprising 15 object classes, with 2,309 samples allocated for training and 581 for testing. To create ScanObjectNN-C, we follow the methodology outlined in (Mirza *et al.*, 2023), applying 15 corruption types exclusively to the test set and focusing on object-level perturbations.

2.4.3 Main Results

Since the pre-trained weights of BFTT3D (Wang *et al.*, 2024c) were not available, we reproduced their results using their code and original settings, along with the same pre-trained weights employed in our method.

ScanObjectNN-C. Table 2.1 highlights the superiority of PG-SP and PG-SF over existing methods. PG-SP achieves the highest mean accuracy of **49.13%**, outperforming all backpropagation-

Table 2.1 Top-1 Classification Accuracy (%) for all distribution shifts in the ScanObjectNN-C dataset. The mean improvements over Source Only are shown in parentheses. Methods with * are reproduced. TTT methods’ results are taken from (Bahri *et al.*, 2025a)

Method	uni	gauss	backg	impul	upsam	rbf	rbf-inv	den-dec	dens-inc	shear	rot	cut	distort	oclsion	lidar	Mean
Source Only	21.69	18.76	16.70	18.42	22.20	45.96	46.99	71.95	69.36	48.88	35.63	72.98	49.40	6.71	8.78	36.96
Test-Time Training (TTT)																
MATE	27.50	29.40	14.30	22.20	25.60	40.80	42.00	73.70	63.20	45.10	35.30	73.30	45.30	7.10	9.30	36.90 (-0.06)
SMART-PC	27.50	39.10	19.30	21.50	29.80	44.20	48.90	68.30	60.20	49.40	45.40	70.10	49.10	8.40	12.20	39.60 (+2.64)
Test-Time Adaptation (TTA) Using Backpropagation																
TTT-Rot	30.20	34.10	16.20	22.60	22.60	32.40	32.10	45.50	45.00	34.50	29.30	47.80	36.20	7.10	8.10	29.58 (-7.38)
SHOT	29.50	31.60	17.60	24.80	27.20	31.00	32.40	40.70	35.00	30.20	26.60	36.60	29.30	10.50	12.40	27.69 (-9.27)
T3A *	35.11	46.82	13.08	30.64	38.55	50.26	53.18	68.50	63.51	50.60	46.64	69.02	53.53	7.23	10.16	42.46 (+5.50)
TENT *	35.28	49.74	14.46	29.60	38.04	50.43	51.81	71.60	68.85	53.70	44.23	71.43	55.77	7.57	9.29	43.45 (+6.49)
Test-Time Adaptation (TTA) Backpropagation Free																
DUA	30.50	40.10	10.20	23.60	29.90	43.70	46.10	68.30	66.30	48.50	38.90	68.70	48.40	8.60	8.10	38.66 (+1.70)
BFTT3D *	16.35	14.97	14.29	17.04	13.43	38.90	42.00	69.71	70.74	41.14	28.92	69.19	43.20	6.54	8.61	33.00 (-3.96)
PG-SP (Ours)	41.48	55.59	22.89	38.90	42.17	56.80	62.31	76.08	76.42	59.04	49.74	76.76	62.48	7.57	8.78	49.13 (+12.17)
PG-SF (Ours)	40.96	54.73	21.86	36.49	39.59	55.25	58.35	75.56	74.01	56.80	48.71	75.56	58.69	7.06	9.47	47.54 (+10.58)

Table 2.2 Top-1 Classification Accuracy (%) for all distribution shifts in the ModelNet-C dataset. The mean improvements over Source Only are shown in parentheses. Methods with * are reproduced. TTT methods’ results are taken from (Bahri *et al.*, 2025a)

Method	uni	gauss	backg	impul	upsam	rbf	rbf-inv	den-dec	dens-inc	shear	rot	cut	distort	oclsion	lidar	Mean
Source Only	66.53	59.12	7.21	31.69	74.72	67.71	69.77	62.28	75.08	74.35	38.13	58.63	70.02	38.53	18.64	54.16
Test-Time Training (TTT)																
MATE	69.80	61.80	18.90	63.90	72.50	64.00	66.00	74.00	80.80	71.00	36.70	69.20	66.30	38.40	29.90	58.90 (+3.99)
SMART-PC	82.40	80.10	12.00	67.10	84.50	76.00	78.60	67.30	72.90	73.30	43.90	72.60	73.50	37.40	24.80	63.10 (+8.94)
Test-Time Adaptation (TTA) Using Backpropagation																
TTT-Rot	61.30	58.30	34.50	48.90	66.70	63.60	63.90	59.80	68.60	55.20	27.30	54.60	64.00	40.00	29.10	53.05 (-1.11)
SHOT	29.60	28.20	9.80	25.40	32.70	30.30	30.10	30.90	31.20	32.10	22.80	27.30	29.40	20.80	18.60	26.60 (-27.56)
T3A *	64.06	61.55	20.14	50.12	64.83	63.74	65.03	66.57	72.29	68.03	50.49	64.67	62.80	43.64	40.28	57.22 (+3.06)
TENT *	73.78	70.14	18.48	59.68	76.14	70.58	72.08	73.22	79.25	73.95	54.25	70.99	70.50	46.07	43.80	63.53 (+9.37)
Test-Time Adaptation (TTA) Backpropagation Free																
DUA	65.00	58.50	14.70	48.50	68.80	62.80	63.20	62.10	66.20	68.80	46.20	53.80	64.70	41.20	36.50	54.73 (+0.57)
BFTT3D *	66.53	59.20	7.01	31.85	65.60	67.79	69.81	77.96	85.25	74.35	38.17	74.80	70.02	38.61	30.51	57.16 (+3.00)
PG-SP (Ours)	77.92	73.83	28.12	69.00	80.88	72.93	76.09	81.40	83.59	77.43	56.97	79.58	74.60	48.99	52.67	68.93 (+14.77)
PG-SF (Ours)	78.12	74.47	40.88	66.69	81.73	73.58	75.81	79.09	82.37	77.23	58.19	75.49	73.58	47.85	45.14	68.68 (+14.52)

free and backpropagation-based techniques by **+12.17%** over Source Only. PG-SF achieves **47.54%**, maintaining strong performance while being fully source-free. Notably, PG-SP surpasses DUA by **+10.47%** and BFTT3D by **+16.13%**, demonstrating its effectiveness in mitigating domain shift at the token level. It ranks first in 13 out of 15 perturbations, excelling against *uniform noise*, *gaussian noise*, and *distortion artifacts*. Similarly, PG-SF achieves *state-of-the-art results for a source-free method*, leveraging CLS-token-driven prototype selection without requiring precomputed source statistics.

ModelNet40-C. As shown in Table 2.2, PG-SP and PG-SF achieve the best mean classification accuracies of **68.93%** and **68.68%**, respectively, exceeding Source Only by **+14.77%** and **+14.52%**. Both methods outperform backpropagation-based techniques such as TENT (63.53%) and T3A (57.22%), proving their efficiency without gradient updates. Compared to backpropagation-free approaches, PG-SP improves over BFTT3D by **+11.77%** and DUA by **+14.20%**, reinforcing the strength of our **token-purging mechanism**. Our method achieves the highest accuracy in 13 out of 15 corruptions, particularly excelling in *background corruption*, *uniform noise*, and *LiDAR distortions*, which typically degrade model performance.

ShapeNet-C. Table 2.3 demonstrates the effectiveness of PG-SP and PG-SF in the *challenging* ShapeNet-C dataset, where all other methods perform worse than Source Only. PG-SF achieves **64.40%** and PG-SP **64.17%**, marking the **only improvements** over Source Only, by **(+3.05%** and **+2.82%)**. Both surpass TENT (54.98%) and T3A (54.18%) while maintaining computational efficiency. Compared to backpropagation-free methods, PG-SP and PG-SF outperform DUA (60.77%) and BFTT3D (60.73%) by over **+3.4%**. PG-SP achieves the best accuracy in noise-based corruptions, particularly *impulse noise*, while PG-SF excels in *background and rotation variations*. Both variants maintain competitive accuracy across density and geometric transformations, reinforcing their robustness to diverse real-world distribution shifts.

Mamba Backbone. Table 2.4 demonstrates that our PG-SP token-purging strategy transfers seamlessly to the state-space *Point-Mamba* (Liang *et al.*, 2024). On **ScanObjectNN-C**, PG-SP lifts the mean accuracy from **29.11%** to **41.42%** (**+12.31%**) and ranks first in 13 of 15 corruptions, with large jumps for *uniform noise*, *Gaussian noise*, and *background*. The gains persist on **ModelNet-C**, where PG-SP attains **68.61%** mean accuracy—**+10.29%** above the source model—while surpassing it on every single perturbation, with the biggest margins on *background* (**+40.5%**), *impulse noise* (**+32.4%**), and the *cut* corruption (**+46.6%**). Although **ShapeNet-C** remains the most challenging benchmark, PG-SP maintains an overall score of **51.66%** (-0.26% versus Source Only) and still alleviates severe shifts such as *background* (**+27.9%**) and *impulse noise* (**+17.2%**).

Table 2.3 Top-1 Classification Accuracy (%) for all distribution shifts in the ShapeNet-C dataset. The mean improvements over Source Only are shown in parentheses. Methods with * are reproduced.

TTT methods’ results are taken from (Bahri *et al.*, 2025a)

Method	uni	gauss	backg	impul	upsam	rbf	rbf-inv	den-dec	dens-inc	shear	rot	cut	distort	oclsion	lidar	Mean
Source Only	77.34	71.78	8.64	54.51	77.90	75.50	76.06	85.25	76.42	80.48	57.08	85.12	76.05	10.97	7.13	61.35
Test-Time Training (TTT)																
MATE-Standard	77.80	74.70	4.30	66.20	78.60	76.30	75.30	86.10	86.60	79.20	56.10	84.10	76.10	12.30	13.10	63.10 (+1.75)
SMART-PC-Standard	80.80	78.90	8.90	60.40	81.80	81.10	81.70	84.80	78.40	80.80	63.70	84.90	79.80	11.50	8.80	64.40 (+3.05)
Test-Time Adaptation (TTA) Using Backpropagation																
TTT-Rot	74.60	72.40	23.10	59.90	74.90	73.80	75.00	81.40	82.00	69.20	49.10	79.90	72.70	14.00	12.00	60.93 (-0.42)
SHOT	44.80	42.50	12.10	37.60	45.00	43.70	44.20	48.40	49.40	45.00	32.60	46.30	39.10	6.20	5.90	36.19 (-25.16)
T3A *	66.67	64.25	13.02	59.77	67.02	65.14	64.89	70.40	66.67	65.98	54.96	69.43	63.79	10.07	10.59	54.18 (-7.17)
TENT *	68.50	65.29	7.55	55.89	67.80	65.77	66.30	76.83	68.00	69.36	51.20	76.35	64.77	10.66	10.36	54.98 (-6.37)
Test-Time Adaptation (TTA) Backpropagation Free																
DUA	76.10	70.10	14.30	60.90	76.20	71.60	72.90	80.00	83.80	77.10	57.50	75.00	72.10	11.90	12.10	60.77 (-0.58)
BFTT3D *	72.58	65.91	10.57	60.86	67.78	73.45	74.38	84.55	83.52	78.58	57.96	83.71	75.17	11.95	10.00	60.73 (-0.62)
PG-SP (Ours)	78.63	76.52	15.78	72.70	78.79	76.49	77.18	83.28	81.18	79.05	63.36	83.07	76.33	9.85	10.34	64.17 (+2.82)
PG-SF (Ours)	79.21	75.43	22.46	68.38	78.95	76.78	77.12	83.52	80.73	79.24	64.91	83.55	76.08	10.15	9.55	64.40 (+3.05)

Table 2.4 Top-1 Classification Accuracy (%) on Mamba networks for all distribution shifts in ScanObjectNN-C, ModelNet-C, and ShapeNet-C datasets. The mean improvements over Source Only are shown in parentheses. The Mamba backbone is from (Liang *et al.*, 2024)

Method	uni	gauss	backg	impul	upsam	rbf	rbf-inv	den-dec	dens-inc	shear	rot	cut	distort	oclsion	lidar	Mean
ScanObjectNN-C																
Source Only	17.73	18.59	10.67	11.19	18.93	32.87	35.11	59.90	49.05	36.32	27.71	62.48	35.80	9.64	10.67	29.11
PG-SP (Ours)	32.29	44.79	25.00	33.85	35.07	45.83	46.53	69.62	63.37	50.35	41.15	67.71	50.00	6.94	8.85	41.42 (+12.31)
ModelNet-C																
Source Only	71.35	64.59	5.35	39.26	78.53	67.30	69.90	79.94	85.62	68.44	35.78	76.66	68.44	36.35	27.35	58.32
PG-SP (Ours)	76.79	73.05	45.82	71.71	79.55	72.36	74.23	84.42	85.02	69.44	49.31	82.43	71.23	46.83	47.04	68.61 (+10.29)
ShapeNet-C																
Source Only	67.20	61.20	4.93	38.79	67.31	66.55	68.06	80.19	59.26	70.26	37.45	80.93	65.96	5.96	4.71	51.92
PG-SP (Ours)	62.89	52.26	32.82	56.04	64.38	62.41	64.18	70.82	57.09	62.37	41.25	69.54	61.38	8.09	9.35	51.66 (-0.26)

2.4.4 Ablation Study

Effect of Purge Size $\mathcal{L}_{pg}$ on the Performance. The purge size $\mathcal{L}_{pg}$ is the sole hyperparameter in PG, controlling the number of tokens removed before they enter the attention layers. Figure 2.3 presents the impact of varying $\mathcal{L}_{pg}$, where the black curve represents a dense evaluation across values from 0 to 127, spanning 127 independent runs. Additionally, the red curve depicts the corresponding average entropy of the logits for each run. As the purge size increases ($0 < L_{pg} < 100$), both TOP-1 ACCs steadily improve, and logit entropy decreases, demonstrating that removing highly corrupted tokens enhances model confidence and overall adaptation performance. Excessive purging ($L_{pg} \gtrsim 100$) causes the removal of clean, informative tokens, resulting in a performance drop along with an increase in entropy. The green dashed line

in Figure 2.3 indicates the maximum accuracy attainable within the dense exploration of different purge sizes in the range of $[0, 32]$. However, our entropy-based selection strategy (the orange dashed line) achieves a close to perfect accuracy while operating over a sparse set of only six purge sizes, $\mathcal{L}_{pg} = \{0, 2, 4, 8, 16, 32\}$. These results validate the effectiveness of our entropy-based approach in dynamically selecting a near-optimal L_{pg} , ensuring robust adaptation without requiring exhaustive hyperparameter tuning. This evaluation is conducted on the *Background* corruption from the ScanObjectNN-C dataset.

Effect of BatchNorm Reset.

We have ablated the effect of resetting BatchNorm statistics on our PG-SP method and our baseline, BFTT3D. The results indicate that resetting BatchNorm statistics consistently improves adaptation performance. Specifically, our PG-SP model achieves **49.13%** accuracy with BatchNorm reset compared to **39.86%** without it, showing a substantial performance gain of (+9.27%). A similar trend is observed for BFTT3D, where BatchNorm reset improves accuracy

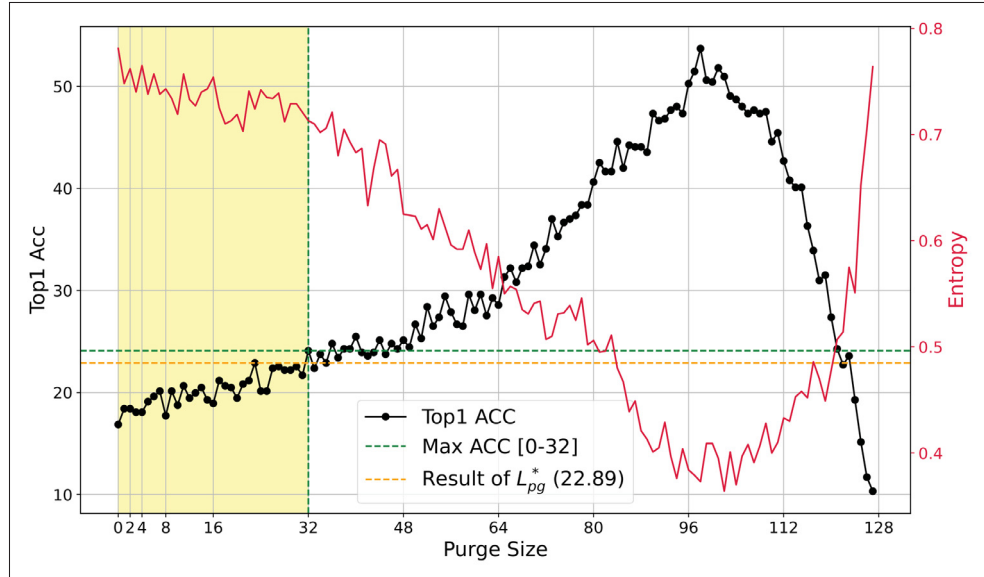


Figure 2.3 Effect of purge size on ACC of **Background** corruption from ScanObjectNN-C dataset

from **33.00%** to **45.76%** (+12.76%). Notably, in both cases, our PG-SP method outperforms BFTT3D, achieving a higher accuracy with and without BatchNorm reset.

Computational Cost. Our PG-SP method demonstrates significant computational efficiency compared to BFTT3D in both execution time and memory usage. `glspg-sp` processes a batch in just **14.07 ms**, whereas BFTT3D requires **173.94 ms**, achieving a **12.4×** speedup. For the sake of fairness, we run our Test-Time Hyperparameter Selection in sequential mode and not in parallel. Additionally, running PG-SP consumes **1300 MiB** of GPU memory in total (backbone and data flow), while BFTT3D requires **7400 MiB**, resulting in a **5.5×** lower memory footprint.

For more detailed ablations, please refer to Appendix I.

2.5 Conclusions

Purge-Gate (PG) is a backpropagation-free test-time adaptation method that removes tokens most affected by domain shifts, ensuring robust inference for 3D point cloud classification. Our approach outperforms existing TTA methods, achieving higher accuracy (+10.3%), 12.4× faster inference, and 5.5× lower memory usage. Extensive evaluations demonstrate that token-level purging is a powerful and efficient alternative to traditional fine-tuning, free from backpropagation. Eliminating the need for iterative updates, PG suits real-time adaptation under domain shift for real-world applications.

CHAPTER 3

REVISITING LAYER NORMALIZATION FOR POINT CLOUD TEST TIME ADAPTATION

Moslem Yazdanpanah* , Ali Bahri , Mehrdad Noori , Sahar Dastani , David Osowiechi ,
Gustavo Adolfo Vargas Hakim , Ismail Ben Ayed , Christian Desrosiers

Department of Software Engineering and IT, École de Technologie Supérieure,
1100 Notre-Dame West, Montreal, Quebec, Canada H3C 1K3

Paper accepted in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, March 2026

Abstract

We analyze Layer Normalization (LN) from a domain (batch) perspective and explain why BatchNorm-style test-time fixes often fail on Transformer backbones. As feature dimension and batch size grow, the per-feature batch marginals after LN’s pre-affine step concentrate at mean ≈ 0 and variance ≈ 1 , making cross-batch re-standardization unnecessary and often harmful. This yields a simple rule: keep the pre-affine LN intact and adjust only the post-affine mean and gain. We instantiate this with **LN-TTA**, a backpropagation-free and source-free, Test-Time Adaptation (TTA) that performs a single forward pass and uniformly reparameterizes each LN layer. On three corrupted 3D point-cloud suites (ScanObjectNN-C, ModelNet40-C, ShapeNet-C), LN-TTA improves over Source-Only by +12.35, +15.58, and +3.03 points, surpasses backpropagation baselines (e.g., TENT), and sustains up to 93 samples/s, on average 39 $\times$ faster and 5 $\times$ more memory-efficient than the next-best backprop-free method. Code is available at: github.com/MosyMosy/LN_TTA.

3.1 Introduction

Transformers have become the default backbone across vision, and point-cloud recognition increasingly relies on token-centric, Transformer-style pipelines. As these models move from training to deployment, however, their accuracy can deteriorate when the test distribution differs

from that encountered in training. Test-Time Adaptation (TTA) addresses this problem by adapting a source-pretrained model during inference using only unlabeled test data (Gandelsman *et al.*, 2022; Gao *et al.*, 2023; Iwasawa & Matsuo, 2021; Lim *et al.*, 2023; Yeo *et al.*, 2023; Schneider *et al.*, 2020). Alongside lightweight transfer strategies (Yazdanpanah & Moradi, 2022), a rich toolbox has emerged for 2D images (Osowiechi *et al.*, 2024a; Noori *et al.*, 2025), including entropy-based updates of batch statistics and affine parameters (Nado *et al.*, 2020; Wang *et al.*, 2020), Laplacian-style clustering objectives (Boudiaf *et al.*, 2022), and multi-augmentation prediction averaging (Zhang *et al.*, 2022b). Yet directly porting these ideas to point clouds is nontrivial: test-time perturbations can change the number of points, distort local geometry in highly non-uniform ways, and interact with the non-Euclidean structure and extreme sparsity of 3D data. Consequently, image-centric adaptation strategies, designed for fixed-resolution arrays, often fail to respect the tokenization and computation patterns of Transformer backbones operating on irregular point sets.

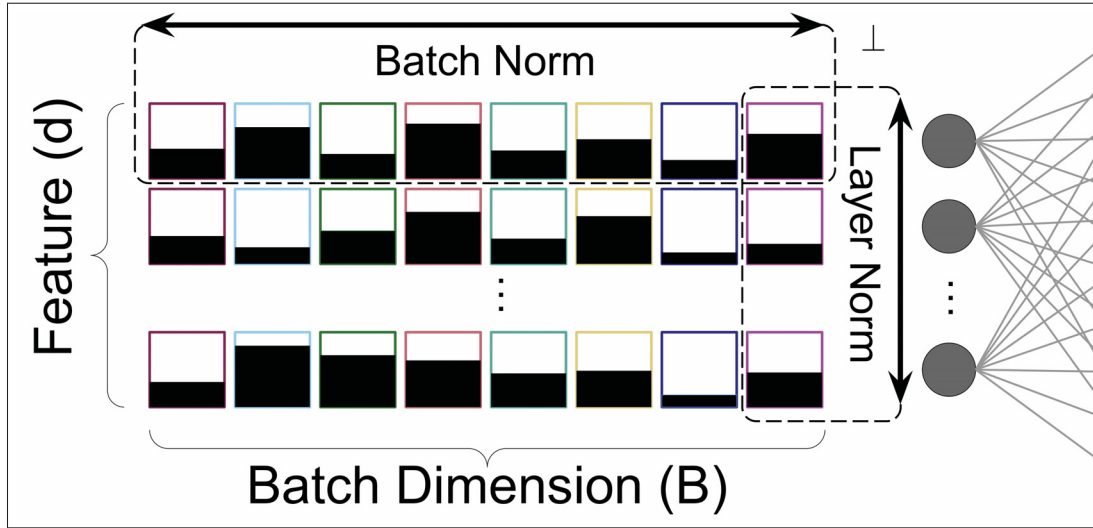


Figure 3.1 **BN vs. LN: axes and TTA levers.** BN normalizes *per-feature across the batch*, stabilizing per-feature marginals; LN normalizes *per-token across features*, removing common mode/rescaling while batch-agnostic. These axes yield different test-time levers: BN updates batch statistics, whereas LN minimally recalibrates post-affine (γ, β). Squares denote activations; colors index samples

Motivated by these differences, several TTA methods have been developed specifically for point-cloud classification (Wang *et al.*, 2024c; Mirza *et al.*, 2023; Wang *et al.*, 2024d; Bahri

et al., 2024b). While encouraging, important gaps remain. The dominant image-TTA baseline TENT (Wang *et al.*, 2020) underperforms on point clouds; MATE (Mirza *et al.*, 2023) couples pretraining and TTA via an MAE reconstruction objective, placing it within TTT and preventing application to arbitrarily pretrained models; and (Bahri *et al.*, 2024b) relies on multiple point-subsampling augmentations, substantially increasing computational cost. Moreover, many approaches iterate several backpropagation steps during inference, which hinders runtime-critical deployments and introduces sensitivity to the update budget. BFTT3D (Wang *et al.*, 2024c) is backpropagation-free but depends on detailed class-level source prototypes, effectively tying adaptation to source data at test time.

A central reason these trade-offs persist is a mismatch between the normalization assumptions behind many domain-adaptation recipes and the normalization actually used in modern Transformer backbones. In the context of domain adaptation, Batch Normalization (BN) is well-understood and widely exploited (Li *et al.*, 2016; Li, Jiang, Zhang, Kamp & Dou, 2021; Zhang, Qi, Shi & Gao, 2023a): by normalizing *across the batch/domain*, BN stabilizes each feature’s batch statistics (means and variances near $(0, 1)$), so every downstream weight sees a steady signal. Transformers, however, rely on Layer Normalization (LN), which normalizes *within each sample across features* and does not use the batch axis. This distinction (illustrated in Figure 3.1) suggests that BN-style strategies cannot be fully transferred to LN-based backbones; instead, a *token-centric* view of domain shift is needed to understand what should be adapted, where, and how.

We take exactly this view. From a domain (batch) perspective, we analyze how LN shapes the inputs seen by downstream layers in Transformer pipelines. Our analysis shows two key effects that clarify the surprising empirical success of LN-based architectures under shift: (i) LN *exactly* removes per-sample shift and global scaling (a projection view), yielding a stable overall scale; and (ii) as the feature dimension d and batch size B grow, the per-feature batch marginals of the *pre-affine* outputs *concentrate* near $(0, 1)$, providing an *implicit* stabilization that explains when BN-style fixes transfer, and when they do not. This understanding leads to a simple, practical design rule for LN-based Transformers: rather than re-standardizing channels

(as in BN), apply minimal *post-affine* recenter/rescale adjustments to the LN affine parameters to make the downstream layers see training-like inputs.

Building on this LN-aware perspective, we develop TTA mechanisms tailored to Transformer backbones on point clouds: methods that are compatible with token-centric inference, avoid dependence on source data, and eliminate backpropagation at test time, while remaining efficient and robust under the diverse, structure-altering shifts encountered in real 3D deployments.

Our contributions can be summarized as follows:

- We formalize the effect of LN on batch-wise stabilization of features. We show that, as feature dimension and batch size grow, LN’s the average of pre-affine outputs concentrates to zero-mean/unit-variance *on the batch (domain) dimension*, stabilizing the input to the subsequent layers.
- Guided by this theory, we introduce a backpropagation-free, source-free, stateless procedure that simply recalibrates each LN’s post-affine parameters using a lightweight prototype, with a single forward pass.
- We conduct extensive experiments to test out our theoretical understanding of LN, as well as evaluate our TTA method on standard TTA benchmarks. Our method reaches State Of The Art (SOTA) performance while being more efficient than the latest SOTA.

3.2 Related Works

Test-Time Adaptation (TTA) for 3D Point Cloud Processing. Extending TTA from 2D to 3D introduces geometry- and ordering-specific challenges, motivating specialized designs. Classical TTA includes TENT (BN affine updates + entropy minimization) (Wang *et al.*, 2020), SHOT (entropy minimization + diversity regularization) (Liang *et al.*, 2020), DUA (BN-statistics update) (Mirza *et al.*, 2022), and T3A (backprop-free classifier refinement via pseudo-prototypes) (Iwasawa & Matsuo, 2021). These methods avoid full retraining but largely hinge on BN-centric or classifier-only adjustments.

One of the earliest Test-Time Training (TTT) approaches tailored for 3D data, MATE (Mirza *et al.*, 2023), leverages a Masked Autoencoder (MAE) reconstruction loss to improve the robustness of point cloud classification models under distribution shifts. Another paper in TTT, SMART-PC (Bahri *et al.*, 2025a), proposes a skeleton-based framework for 3D point cloud classification that enhances robustness to distribution shifts by predicting skeletal representations during pre-training to capture corruption-invariant geometric features.

Diffusion-based TTA, CloudFixer (geometric transform optimization with a pre-trained diffusion model), and DDA (projecting test inputs toward the source via a generative diffuser with guidance/self-ensembling), improve robustness without retraining (Shim, Kim & Yang, 2024; Gao *et al.*, 2023). A complementary 3D approach leverages sampling variation with weight-averaged ensembles (Bahri *et al.*, 2024b). However, iterative denoising, per-sample optimization, and multiple passes make these methods compute- and latency-heavy, limiting real-time or resource-constrained deployment.

BFTT3D (Wang *et al.*, 2024c) proposes a backpropagation-free TTA approach tailored for 3D point clouds, employing a two-stream architecture to preserve information from both source and target domains. Nevertheless, its dependence on class-specific prototypes from the source data limits its applicability in fully source-free TTA scenarios. Purge-Gate (Yazdanpanah *et al.*, 2025) introduces a backpropagation-free TTA method that discards tokens most affected by domain shifts before they reach the attention layers, enabling robust adaptation without iterative updates. Purge-Gate includes a source-aware variant (PG-SP) and a fully source-free version (PG-SF), both achieving state-of-the-art accuracy and good speed and memory efficiency gains on different datasets.

Normalization layers. Layer Normalization (LN) normalizes features *within* each sample, removing the mini-batch dependence of BatchNorm (BN) (Ba *et al.*, 2016). Beyond stabilization, theory shows LN’s benefits arise from gradients through mean/variance paths (Xu, Sun, Zhang, Zhao & Lin, 2019), scale-invariance that reduces sharpness (Lyu *et al.*, 2022), and a spherical optimization geometry (Roburin *et al.*, 2022); it can be decomposed into mean removal, radius

normalization in the orthogonal subspace, and a per-feature affine, with mean removal often redundant at inference (Gupta, Ozdemir & Anumanchipalli, 2024). In Transformers, placement matters: Pre-LN stabilizes training compared to Post-LN (Xiong *et al.*, 2020), while DeepNorm modifies residual scaling to enable ultra-deep models (Wang *et al.*, 2024b). Lighter variants such as RMSNorm drop mean-centering yet retain accuracy with lower overhead (Zhang & Sennrich, 2019); hybrid schemes (Switchable Norm, BLN) learn or interpolate statistics to adapt across batch regimes (Luo, Zhang, Ren, Peng & Li, 2019; Ziaee & Çano, 2022). Most TTA updates exploit BN statistics (e.g., TENT) (Wang *et al.*, 2020), but modern LN/RMSNorm backbones lack reliable batch re-estimation. This motivates *affine-only* LN edits at inference, adjusting common mode and global scale while preserving token geometry, guided by lightweight proxies (e.g., [cls]-based summaries) (Zou, Yi, Li & Li, 2024a). Despite the valuable contributions to the field, most of these methods are based on empirical evaluations; hence, we base our study on a theoretical analysis of LN layers in the domain, and then we extend our understanding to a forward-only, stateless post-affine recalibration tailored to LN. We achieve this by aligning target activations with the source-trained regime without backpropagation or access to source data.

3.3 Method

In this section, we analyze LayerNorm as a geometric projection-and-rescaling operator (Section 3.3.2). Building on this view, we derive exact token-wise identities and batch-wise concentration results for the pre- and post-affine statistics of LayerNorm (Section 3.3.3), showing that normalization intrinsically stabilizes certain quantities while leaving others under the control of the affine parameters. These insights lead directly to our backpropagation-free strategy, LN Test Time Adaptation (LN-TTA), which preserves the projection-and-normalization steps and performs a lightweight post-affine reparameterization using either source moments or a single stored prototype (Section 3.3.4).

3.3.1 Preliminaries

A point cloud is an unordered set of 3D points

$$X_i = \{p_{i,1}, p_{i,2}, \dots, p_{i,N_p}\} \subset \mathbb{R}^3, \quad (3.1)$$

where N_p is the number of points.

Following (Pang *et al.*, 2022), each sample X_i is tokenized by selecting N token centers (typically $N \ll N_p$) via farthest point sampling (FPS) and forming neighborhoods around them. We initialize $c_1 \in X_i$ (arbitrary) and, for $t = 2, \dots, N$,

$$c_t = \arg \max_{p \in X_i} \min_{1 \leq s < t} \|p - c_s\|_2, \quad (3.2)$$

and define neighborhoods via k -nearest neighbors:

$$\mathcal{N}_t := \text{KNN}_k(c_t; X_i). \quad (3.3)$$

A token embedding map $\mathcal{E}$ produces d -dimensional token vectors, which we stack row-wise:

$$x_i = \begin{bmatrix} \mathcal{E}(c_1, \mathcal{N}_1)^\top \\ \vdots \\ \mathcal{E}(c_N, \mathcal{N}_N)^\top \end{bmatrix} \in \mathbb{R}^{N \times d}, \quad x_{i,t} \equiv x_i[t] \in \mathbb{R}^d. \quad (3.4)$$

A Transformer backbone processes x_i and a classifier head produces logits $\hat{y}_i = f_\theta(x_i)$.

The labeled *source* training set and unlabeled *target* test set are

$$\mathcal{D}_S = \{(x_i^S, y_i)\}_{i=1}^{N_S}, \quad \mathcal{D}_T = \{x_j^T\}_{j=1}^{N_T}, \quad (3.5)$$

with $P_S(x) \neq P_T(x)$.

Inside the Transformer, each LayerNorm (LN) (Ba *et al.*, 2016) acts *per token* (row-wise across features). For any token $x_t \in \mathbb{R}^d$,

$$\text{LN}(x_t) = \gamma \odot \frac{x_t - \mu_d(x_t)\mathbf{1}}{\sqrt{\sigma_d^2(x_t) + \varepsilon}} + \beta, \quad (3.6)$$

where $\gamma, \beta \in \mathbb{R}^d$ are learnable affine parameters, $\odot$ is elementwise multiplication, $\varepsilon > 0$ is the LN epsilon, and $\mathbf{1} \in \mathbb{R}^d$ is the all-ones vector. The per-token mean and variance are

$$\mu_d(x_t) = \frac{1}{d} \sum_{j=1}^d (x_t)_j, \quad \sigma_d^2(x_t) = \frac{1}{d} \sum_{j=1}^d ((x_t)_j - \mu_d(x_t))^2. \quad (3.7)$$

We also write

$$\bar{m}(\beta) := d^{-1} \mathbf{1}^\top \beta, \quad \bar{s}(\gamma) := (d^{-1} \|\gamma\|_2^2)^{1/2} \quad (3.8)$$

for the mean and root-mean-square of the affine parameters.

Dimensions and indices. Unless stated otherwise, $x \in \mathbb{R}^{B \times N \times d}$ denotes a batch of token sequences, where B is batch size, N is number of tokens per sample, and d is embedding dimension. We use i for sample index, $t \in \{1, \dots, N\}$ for token index, and $b \in \{1, \dots, B\}$ for batch index. When invoking the limits in Theorem 3.3.3, we fix a token index t and a feature index j .

As LN standardizes each token across its d features, it is blind to batch-wise (domain) structure, yet it can strongly shape the distribution seen by downstream layers. Methods like TENT (Wang *et al.*, 2020) have been ported from BatchNorm (BN) to LN in Transformers, but often without a theoretical account of the different normalization axes (features for LN vs. batch for BN; see Figure 3.1). We therefore analyze LN formally and, based on this understanding, propose LN-TTA: a backpropagation-free update of the LN *affine* that cancels domain shift without re-standardizing channels or updating normalization buffers.

3.3.2 Geometric (Projection) View of LN

To facilitate analysis, we adopt a geometric view of LN that decomposes it into three steps: (i) project onto the mean-zero hyperplane, (ii) normalize the radius, and (iii) apply a per-feature affine. This clarifies what LN changes, what it leaves invariant, and which degrees of freedom are suitable for adaptation.

Let $\mathbf{u} = \mathbf{1}/\sqrt{d}$,

$$\mathbf{C} = I - \mathbf{u}\mathbf{u}^\top, \quad \mathcal{H} = \{v \in \mathbb{R}^d : \mathbf{u}^\top v = 0\}. \quad (3.9)$$

Then $\mathbf{C}$ removes the token mean: $\mathbf{C}x_t = x_t - \mu_d(x_t)\mathbf{1} \in \mathcal{H}$, and

$$\mu_d(x_t) = \frac{1}{\sqrt{d}}\mathbf{u}^\top x_t, \quad \sigma_d^2(x_t) = \frac{1}{d}\|\mathbf{C}x_t\|^2. \quad (3.10)$$

The pre- and post-affine maps can be written as

$$\begin{aligned} \widehat{\text{LN}}(x_t) &= \sqrt{d} \frac{\mathbf{C}x_t}{\sqrt{\|\mathbf{C}x_t\|^2 + d\varepsilon}}, \\ \text{LN}(x_t) &= \Gamma \widehat{\text{LN}}(x_t) + \beta, \quad \Gamma = \text{diag}(\gamma). \end{aligned} \quad (3.11)$$

Thus LN is “project to $\mathcal{H}$ then normalize the radius,” followed by a per-feature affine. For $\varepsilon=0$ and $\sigma_d(x_t)>0$ we obtain

$$\theta = \frac{\mathbf{C}x_t}{\|\mathbf{C}x_t\|} \in \mathcal{H} \cap \mathbb{S}^{d-2}, \quad \widehat{\text{LN}}(x_t) = \sqrt{d} \theta, \quad (3.12)$$

and

$$\|\widehat{\text{LN}}(x_t)\| = \sqrt{\frac{d \sigma_d^2(x_t)}{\sigma_d^2(x_t) + \varepsilon}}. \quad (3.13)$$

If $\mathbf{C}x_t=0$, then for $\varepsilon>0$ we have $\widehat{\text{LN}}(x_t) = 0$ and the post-affine output equals β (If $\mathbf{C}x_t = 0$, the pre-affine direction in Equation (3.11) is undefined for $\varepsilon = 0$; for $\varepsilon > 0$ we have $\widehat{\text{LN}}(x_t) = 0$ and the post-affine output equals β).

We will reuse $\mathbf{C}$, $\mathcal{H}$ and the identities in (3.11)–(3.13) in Section 3.3.3 and as the basis for our post-affine adaptation in Section 3.3.4.

3.3.3 Effect on Mean and Variance: Token-wise Identities and Batch-wise Limits

Building on the projection view, we first characterize what the pre-affine map $\widehat{\mathbf{LN}}$ guarantees for means and variances per token, independently of (γ, β) , and then analyze how these statistics behave across a batch.

We begin with invariances of $\widehat{\mathbf{LN}}$ to additive common-mode shifts and global rescaling.

Lemma 3.3.1 (Pre-affine invariances). *Assume either $\sigma_d(x_t) > 0$ or $\varepsilon > 0$. For any $b \in \mathbb{R}$, $\widehat{\mathbf{LN}}(x_t + b\mathbf{1}) = \widehat{\mathbf{LN}}(x_t)$. For $a > 0$,*

$$\widehat{\mathbf{LN}}(a x_t) = \widehat{\mathbf{LN}}(x_t) f(a; x_t), \quad (3.14)$$

and for $a < 0$,

$$\widehat{\mathbf{LN}}(a x_t) = -\widehat{\mathbf{LN}}(x_t) f(|a|; x_t), \quad (3.15)$$

with

$$f(a; x_t) = \frac{a\sqrt{r + \varepsilon}}{\sqrt{a^2 r + \varepsilon}}, \quad r = d^{-1} \|\mathbf{C}x_t\|^2. \quad (3.16)$$

When $\varepsilon = 0$, magnitudes are exactly scale-invariant and $a < 0$ flips the direction $\theta \mapsto -\theta$.

Next, we state the exact token-wise mean and squared radius produced by $\widehat{\mathbf{LN}}$.

Proposition 3.3.2 (Token-wise exact standardization). *Let $h = \widehat{\mathbf{LN}}(x_t)$. If $\varepsilon = 0$ and $\sigma_d(x_t) > 0$, then $d^{-1} \sum_{j=1}^d h_j = 0$ and $d^{-1} \sum_{j=1}^d h_j^2 = 1$. For $\varepsilon > 0$, the first identity holds and the second equals $\sigma_d^2(x_t) / (\sigma_d^2(x_t) + \varepsilon)$.*

To understand what downstream layers observe across a batch, we adopt a mild concentration setting. We assume that coordinates of $x_{b,t} \in \mathbb{R}^d$ obey a law of large numbers (LLN) across d with uniformly bounded fourth moments. Let (μ, σ^2) denote the per-coordinate population mean

and variance with $\sigma^2 \in (0, \infty)$, and assume $\{x_{b,t}\}_{b=1}^B$ are i.i.d. in b . Under these conditions, we obtain coordinate-wise concentration of the pre-affine outputs.

Theorem 3.3.3 (Asymptotic coordinate standardization). *Fix a feature index j and a token index t . Let $H_{b,t} = \widehat{\text{LN}}(x_{b,t})$ with $\varepsilon=0$. Define the empirical batch statistics*

$$\overline{H}_{j,B} = \frac{1}{B} \sum_{b=1}^B (H_{b,t})_j, \quad S_{j,B}^2 = \frac{1}{B} \sum_{b=1}^B ((H_{b,t})_j - \overline{H}_{j,B})^2. \quad (3.17)$$

Under the assumptions above, as $d \rightarrow \infty$ and $B = B(d) \rightarrow \infty$,

$$\overline{H}_{j,B(d)} \xrightarrow{\mathbb{P}} 0, \quad S_{j,B(d)}^2 \xrightarrow{\mathbb{P}} 1. \quad (3.18)$$

Finally, we propagate these pre-affine limits through the per-feature affine map $y = \Gamma H + \beta$. For any $\varepsilon \geq 0$, the per-feature batch limits become

$$\overline{y}_{j,B(d)} \xrightarrow{\mathbb{P}} \beta_j, \quad S_{y,j,B(d)}^2 \xrightarrow{\mathbb{P}} \gamma_j^2 \frac{\sigma^2}{\sigma^2 + \varepsilon}, \quad (3.19)$$

reducing to (β_j, γ_j^2) at $\varepsilon=0$ (see Theorem 2, in Appendix II). Thus, although LN operates within-token, it induces an *implicit per-feature batch stabilization*: normalized activations concentrate near mean 0 and variance 1, and the post-affine mean/scale are controlled by (β, γ) .

This is the key design rule behind LN-TTA (Section 3.3.4): rather than re-standardizing channels across the batch (as in BN), we keep $\widehat{\text{LN}}$ intact and make minimal post-affine adjustments to (γ, β) so that downstream layers see training-like inputs.

3.3.4 Backpropagation-Free LN-TTA

From Sections 3.3.2 and 3.3.3, we know that the pre-affine LN (i) removes common-mode bias and (ii) normalizes the radius, leaving residual domain variation largely as angular/anisotropic structure within $\mathcal{H}$; in contrast, the post-affine mean and global gain are governed by (γ, β) . Guided by this, LN-TTA leaves the projection-and-normalization unchanged and applies a

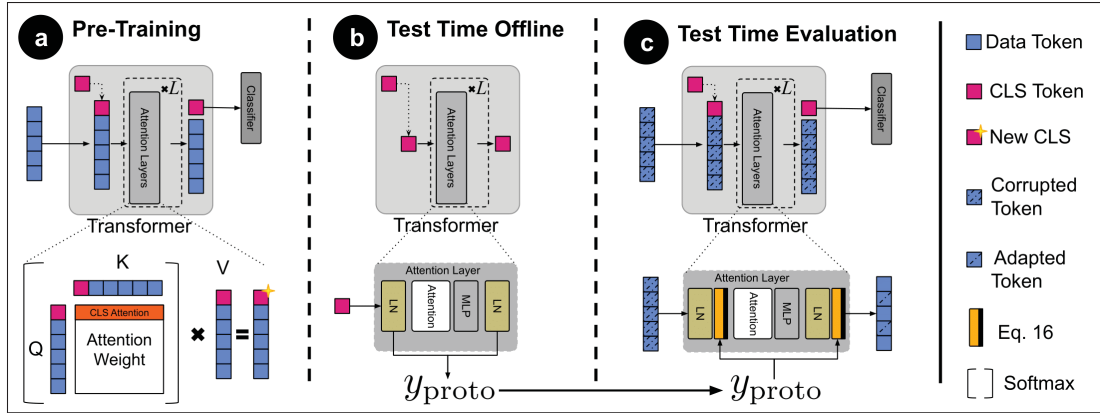


Figure 3.2 **(a) Pre-training.** The [cls] token learns to aggregate domain information as it attends to the training-time token stream. **(b) Prototype extraction.** At test time, with the model frozen, we run a single forward pass using only the pre-trained [cls] token and store layer-wise prototypes for each LN layer. **(c) Test-Time Adaptation (TTA).** During evaluation on corrupted target data, we reparameterize the *post-affine* of each LN layer with the stored prototypes, adapting tokens in a single forward pass (no gradients, no source data)

lightweight post-affine adaptation so that downstream layers receive inputs close to their source-trained regime, while preserving the geometry in $\mathcal{H}$.

We instantiate this idea in two forms, both implemented as non-mutating forward edits:

- a *diagonal moment match* when per-layer source post-affine moments are available;
- a *data-free single-prototype calibration* based on the [cls] token when source data are inaccessible.

Both variants are forward-only, stateless, and $O(d)$ per token.

Let (γ_0^S, β_0^S) denote the LN parameters learned on the source domain $\mathcal{D}_S$, and $\Gamma_0^S = \text{diag}(\gamma_0^S)$.

For any token $x_{b,t} \in \mathbb{R}^d$, define

$$h_{b,t} = \widehat{\text{LN}}(x_{b,t}), \quad y_{b,t} = \Gamma_0^S h_{b,t} + \beta_0^S = \text{LN}(x_{b,t}). \quad (3.20)$$

Over a short *target* window $\mathcal{W}_T \subset \mathcal{D}_T$ containing M tokens (while B remains the mini-batch size), we estimate the *per-feature, across-token* statistics of pre-affine outputs:

$$\widehat{\mu}_h^T = \frac{1}{M} \sum_{(b,t) \in \mathcal{W}_T} h_{b,t}^T, \quad (\widehat{\sigma}_{h,j}^2)^T = \frac{1}{M} \sum_{(b,t) \in \mathcal{W}_T} (h_{b,t,j}^T - \widehat{\mu}_{h,j}^T)^2. \quad (3.21)$$

These can be computed directly from $h_{b,t}^T$ or via

$$h_{b,t}^T = (y_{b,t}^{(T)} - \beta_0^S) \oslash \gamma_0^S, \quad (3.22)$$

where $\oslash$ denotes elementwise division. We use $(\widehat{\sigma}_{h,j}^2)^T + \delta$ in denominators with a small ridge $\delta > 0$ and skip channels with $|\gamma_{0,j}^S| < \tau$ to avoid numerical issues.

If per-layer *source* post-affine moments are available, let

$$\mu_\star^S \in \mathbb{R}^d, \quad \Sigma_\star^S = \text{diag}(\sigma_{*,1}^{2,S}, \dots, \sigma_{*,d}^{2,S}) \quad (3.23)$$

denote the per-feature mean and diagonal variance of post-affine outputs. A diagonal moment match yields

$$\gamma_j^\star = \text{sgn}(\gamma_{0,j}^S) \sqrt{\frac{\sigma_{*,j}^{2,S}}{(\widehat{\sigma}_{h,j}^2)^T + \delta}}, \quad \beta^\star = \mu_\star^S - \Gamma^\star \widehat{\mu}_h^T, \quad (3.24)$$

where $\Gamma^\star = \text{diag}(\gamma^\star)$. The resulting non-mutating forward edit for each target token is

$$y_{\text{adapt}, b, t}^{(T)} = (y_{b,t}^{(T)} - \beta_0^S) \odot (\gamma^\star \oslash \gamma_0^S) + \beta^\star. \quad (3.25)$$

If $|\gamma_{0,j}^S| < \tau$, we set $(\gamma^\star / \gamma_0^S)_j := 1$ to avoid inflating near-zero channels.

In many realistic settings, $(\mu_\star^S, \Sigma_\star^S)$ are unavailable because source data are inaccessible. We then replace them with a single-token proxy obtained from the [cls] token. In pretrained Transformers, [cls] aggregates domain information (Zou *et al.*, 2024a). With the model in eval mode and *no* source or target samples, we run a single forward pass using only the [cls]

embedding and, for each LN layer, record

$$y_{\text{proto}} = \text{LN}([\text{cls}]_{\text{emb}}), \quad m_{\star}^{\text{proto}} = \bar{m}(y_{\text{proto}}), \quad s_{\star}^{\text{proto}} = \frac{\|y_{\text{proto}} - m_{\star}^{\text{proto}} \mathbf{1}\|_2}{\sqrt{d}}. \quad (3.26)$$

This provides a two-scalar proxy for $(\mu_{\star}^S, \Sigma_{\star}^S)$ (see Figure 3.2).

At test time, we apply a fixed, token-wise *recolouring* after each LN affine that preserves the angular structure in $\mathcal{H}$ and adjusts only common mode and global scale:

$$y_{\text{adapt}, b, t}^{(T)} = \frac{s_{\star}^{\text{proto}}}{\bar{s}(\gamma_0^S)} y_{b, t}^{(T)} + \left(m_{\star}^{\text{proto}} - \frac{s_{\star}^{\text{proto}}}{\bar{s}(\gamma_0^S)} \bar{m}(\beta_0^S) \right) \mathbf{1}. \quad (3.27)$$

Under the bounded-fourth-moment/LLN assumptions of Section 3.3.3, token-wise across-feature statistics for the target post-affine outputs concentrate as

$$m(y^{(T)}) = \bar{m}(\beta_0^S) + O_{\mathbb{P}}(d^{-1/2}), \quad s(y^{(T)}) = \bar{s}(\gamma_0^S) + O_{\mathbb{P}}(d^{-1/2}), \quad (3.28)$$

where $m(y) = d^{-1} \mathbf{1}^\top y$ and $s(y) = (d^{-1} \|y - m(y) \mathbf{1}\|_2^2)^{1/2}$. Thus the standardize→recolour procedure reduces (up to $O_{\mathbb{P}}(d^{-1/2})$) to the fixed affine in Equation (3.27) (see Corollary 1.1 in Appendix II), requiring no access to $\mathcal{D}_S$.

Equivalently, Equation (3.27) induces a uniform reparameterization for target-time inference:

$$\gamma'^T = \frac{s_{\star}^{\text{proto}}}{\bar{s}(\gamma_0^S)} \gamma_0^S, \quad \beta'^T = \frac{s_{\star}^{\text{proto}}}{\bar{s}(\gamma_0^S)} \beta_0^S + \left(m_{\star}^{\text{proto}} - \frac{s_{\star}^{\text{proto}}}{\bar{s}(\gamma_0^S)} \bar{m}(\beta_0^S) \right). \quad (3.29)$$

In practice, we simply replace (γ_0^S, β_0^S) by (γ'^T, β'^T) in the forward pass. The procedure is stateless, label-free, source-free, and adds only $O(d)$ work per token. We evaluate LN-TTA in Section 3.4.

3.4 Experiments

All datasets, corruption definitions, evaluation protocols, implementation settings, extra experiments, and ablations are provided in the Appendix II.

Table 3.1 Top-1 Classification Accuracy (%) for all distribution shifts in the ScanObjectNN-C dataset. The mean improvements over Source Only are shown in parentheses. Methods with ● use backpropagation during the adaptation. Methods with ● require accessing the source (training-time) data. Point Cloud Per Second (PPS) represents the method’s throughput. **Bold** values in standard mode show the best values among the **Backpropagation-Free** methods. Results of methods with * are reproduced

Method	uni	gauss	backg	impul	upsam	rbf	rbf-inv	den-dec	dens-inc	shear	rot	cut	distort	oclsion	lidar	PPS	Mean
Source Only	21.69	18.76	16.70	18.42	22.20	45.96	46.99	71.95	69.36	48.88	35.63	72.98	49.40	6.71	8.78	88.78	36.96
Test-Time Training (TTT) Standard																	
● MATE	27.50	29.40	14.30	22.20	25.60	40.80	42.00	73.70	63.20	45.10	35.30	73.30	45.30	7.10	9.30	4.00	36.90 (-0.06)
● SMART-PC	27.50	39.10	19.30	21.50	29.80	44.20	48.90	68.30	60.20	49.40	45.40	70.10	49.10	8.40	12.20	33.00	39.60 (+2.64)
Test-Time Adaptation (TTA) Standard																	
● TTT-Rot	30.20	34.10	16.20	22.60	22.60	32.40	32.10	45.50	45.00	34.50	29.30	47.80	36.20	7.10	8.10	—	29.58 (-7.38)
● SHOT	29.50	31.60	17.60	24.80	27.20	31.00	32.40	40.70	35.00	30.20	26.60	36.60	29.30	10.50	12.40	—	27.69 (-9.27)
● T3A	35.11	46.82	13.08	30.64	38.55	50.26	53.18	68.50	63.51	50.60	46.64	69.02	53.53	7.23	10.16	—	42.46 (+5.50)
● TENT*	35.28	49.74	14.46	29.60	38.04	50.43	51.81	71.60	68.85	53.70	44.23	71.43	55.77	7.57	9.29	40.72	43.45 (+6.49)
Backpropagation-Free																	
AdaBN	31.15	39.59	14.29	36.49	28.92	48.88	53.87	72.29	71.26	51.12	46.13	72.29	55.25	7.92	10.84	14.77	42.69 (+5.73)
DUA	30.50	40.10	10.20	23.60	29.90	43.70	46.10	68.30	66.30	48.50	38.90	68.70	48.40	8.60	8.10	1.83	38.66 (+1.70)
● BFTT3D*	16.35	14.97	14.29	17.04	13.43	38.90	42.00	69.71	70.74	41.14	28.92	69.19	43.20	6.54	8.61	1.67	33.00 (-3.96)
LN-TTA Our	43.20	57.49	25.13	40.28	43.37	56.63	60.59	76.25	74.01	56.28	50.95	77.63	60.59	7.40	9.81	64.24	49.31 (+12.35)
Test-Time Adaptation (TTA) Online																	
● TENT*	38.38	52.67	16.70	35.11	37.87	52.15	55.42	73.84	70.22	53.36	48.37	75.22	58.69	7.23	10.33	40.72	45.70 (+8.74)
LN-TTA Our	43.20	57.49	25.13	40.28	43.37	56.63	60.59	76.25	74.01	56.28	50.95	77.63	60.59	7.40	9.81	64.24	49.31 (+12.35)

Table 3.2 Top-1 Classification Accuracy (%) for all distribution shifts in the ModelNet-C dataset. The mean improvements over Source Only are shown in parentheses. Methods with ● use backpropagation during the adaptation. Methods with ● require accessing the source (training-time) data. Point Cloud Per Second (PPS) represents the method’s throughput. **Bold** values in standard mode show the best values among the **Backpropagation-Free** methods. Results of methods with * are reproduced

Method	uni	gauss	backg	impul	upsam	rbf	rbf-inv	den-dec	dens-inc	shear	rot	cut	distort	oclsion	lidar	PPS	Mean
Source Only	66.53	59.12	7.21	31.69	74.72	67.71	69.77	62.28	75.08	74.35	38.13	58.63	70.02	38.53	18.64	112.41	54.16
Test-Time Training (TTT) Standard																	
● MATE	69.80	61.80	18.90	63.90	72.50	64.00	66.00	74.00	80.80	71.00	36.70	69.20	66.30	38.40	29.90	6.00	58.90 (+3.99)
● SMART-PC	82.40	80.10	12.00	67.10	84.5	76.00	78.60	67.30	72.90	73.30	43.90	72.60	73.50	37.40	24.80	49.00	63.10 (+8.94)
Test-Time Adaptation (TTA) Standard																	
● TTT-Rot	61.30	58.30	34.50	48.90	66.70	63.60	63.90	59.80	68.60	55.20	27.30	54.60	64.00	40.00	29.10	—	53.05 (-1.11)
● SHOT	29.60	28.20	9.80	25.40	32.70	30.30	30.10	30.90	31.20	32.10	22.80	27.30	29.40	20.80	18.60	—	26.60 (-27.56)
● T3A	64.06	61.55	20.14	50.12	64.83	63.74	65.03	66.57	72.29	68.03	50.49	64.67	62.80	43.64	40.28	—	57.22 (+3.06)
● TENT*	73.78	70.14	18.48	59.68	76.14	70.58	72.08	73.22	79.25	73.95	54.25	70.99	70.50	46.07	43.80	69.17	63.53 (+9.37)
Backpropagation-Free																	
AdaBN	79.98	75.65	24.68	67.50	67.34	75.32	76.18	82.01	84.44	78.89	59.89	79.70	75.08	50.32	41.65	27.77	67.91 (+13.75)
DUA	65.00	58.50	14.70	48.50	68.80	62.80	63.20	62.10	66.20	68.80	46.20	53.80	64.70	41.20	36.50	2.07	54.73 (+0.57)
● BFTT3D*	66.53	59.20	7.01	31.85	65.60	67.79	69.81	77.96	85.25	74.35	38.17	74.80	70.02	38.61	30.51	2.20	57.16 (+3.00)
LN-TTA Our	79.86	75.53	29.3	67.99	81.81	76.01	76.42	80.88	83.47	78.32	60.17	77.59	74.88	52.47	51.42	93.03	69.74 (+15.58)
Test-Time Adaptation (TTA) Online																	
● TENT*	75.57	70.62	21.23	61.79	78.61	71.03	72.29	73.14	79.13	75.45	53.97	69.45	70.46	46.84	43.72	69.17	64.22 (+10.06)
LN-TTA Our	79.86	75.53	29.30	67.99	81.81	76.01	76.42	80.88	83.47	78.32	60.17	77.59	74.88	52.47	51.42	93.03	69.74 (+15.58)

Table 3.3 Top-1 Classification Accuracy (%) for all distribution shifts in the ShapeNet-C dataset. The mean improvements over Source Only are shown in parentheses. Methods with ● use backpropagation during the adaptation. Methods with ● require accessing the source (training-time) data. Point Cloud Per Second (PPS) represents the method’s throughput. **Bold** values in standard mode show the best values among the **Backpropagation-Free** methods. Results of methods with * are reproduced

Method	uni	gauss	backg	impul	upsam	rbf	rbf-inv	den-dec	dens-inc	shear	rot	cut	distort	oclsion	lidar	PPS	Mean
Source Only	77.34	71.78	8.64	54.51	77.90	75.50	76.06	85.25	76.42	80.48	57.08	85.12	76.05	10.97	7.13	93.89	61.35
Test-Time Training (TTT) Standard																	
● MATE	77.80	74.70	4.30	66.20	78.60	76.30	75.30	86.1	86.60	79.20	56.10	84.10	76.10	12.30	13.10	5.00	63.10 (+1.75)
● SMART-PC	80.80	78.90	8.90	60.40	81.80	81.10	81.70	84.80	78.40	80.80	63.70	84.90	79.80	11.50	8.80	35.00	64.40 (+3.05)
Test-Time Adaptation (TTA) Standard																	
● TTT-Rot	74.60	72.40	23.10	59.90	74.90	73.80	75.00	81.40	82.00	69.20	49.10	79.90	72.70	14.00	12.00	—	60.93 (-0.42)
● SHOT	44.80	42.50	12.10	37.60	45.00	43.70	44.20	48.40	49.40	45.00	32.60	46.30	39.10	6.20	5.90	—	36.19 (-25.16)
● T3A	66.67	64.25	13.02	59.77	67.02	65.14	64.89	70.40	66.67	65.98	54.96	69.43	63.79	10.07	10.59	—	54.18 (-7.17)
● TENT*	68.50	65.29	7.55	55.89	67.80	65.77	66.30	76.83	68.00	69.36	51.20	76.35	64.77	10.66	10.36	56.90	54.98 (-6.37)
Backpropagation-Free																	
AdaBN	71.72	69.33	8.11	65.48	58.90	75.09	75.02	82.90	81.97	78.12	62.48	82.38	74.94	9.97	10.20	25.77	60.44 (-0.91)
DUA	76.10	70.10	14.30	60.90	76.20	71.60	72.90	80.00	83.80	77.10	57.50	75.00	72.10	11.90	12.10	1.71	60.77 (-0.58)
● BFTT3D*	72.58	65.91	10.57	60.86	67.78	73.45	74.38	84.55	83.52	78.58	57.96	83.71	75.17	11.95	10.00	1.90	60.73 (-0.62)
LN-TTA Our	79.18	76.62	14.65	68.72	79.28	77.95	78.20	84.38	82.32	79.43	65.25	83.91	77.12	9.11	9.41	73.12	64.38 (+3.03)
Test-Time Adaptation (TTA) Online																	
● TENT*	78.58	74.91	13.10	67.18	78.49	76.19	76.23	83.55	79.11	78.11	62.90	82.76	74.85	9.42	9.58	56.90	63.00 (+1.65)
LN-TTA Our	79.18	76.62	14.65	68.72	79.28	77.95	78.20	84.38	82.32	79.43	65.25	83.91	77.12	9.11	9.41	73.12	64.38 (+3.03)

3.4.1 Results

3.4.1.1 LN Analyses

Memory footprint. We additionally measure peak GPU memory under the common evaluation setup (batch size 32). Our LN-TTA uses only **1.468 GB** similar to Source Only, whereas TENT requires **2.890 GB** ($\sim 2\times$ more), BFTT3D **7.443 GB** ($\sim 5\times$), and DUA **76.000 GB** ($\sim 52\times$); see Figure 3.3. Because LN-TTA is a purely forward, stateless post-LN reparameterization, it stores neither a backward graph nor optimizer state, BN buffers, or feature banks, explaining its low footprint. This efficiency enables larger batches or higher-resolution tokenizations within the same memory budget and complements the accuracy/throughput gains in Tables 2.2, 2.3 and 3.1.

Concentration vs. batch and dimensionality. We quantify how close the post-LN activations are to a normal distribution. We sweep batch size B and feature dimension d over a log-spaced grid, computing per-feature histograms across batch after LN, averaging these histograms over features, and measuring the divergence to the ideal $\mathcal{N}(0, 1)$ using Jensen–Shannon (JS) distance. The resulting contour map in Figure 3.4 shows that this distance shrinks monotonically as either

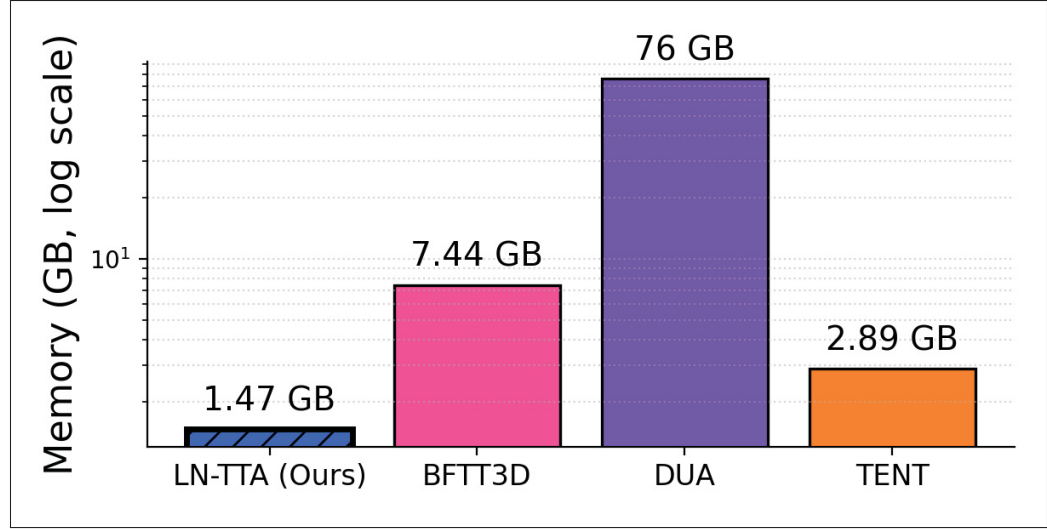


Figure 3.3 Average of GPU memory (log scale; lower is better). The reported numbers are for total GPU memory (Backbone + Data) with $B = 32$

B or d increases, with especially rapid improvement as d grows. These trends match our analysis in Section 3.3.2–Section 3.3.3: LN enforces exact token-wise zero mean and unit variance (Theorem 3.3.2), and the per-feature, across-tokens marginals concentrate near $(0, 1)$ as (d, B) grow (Theorem 3.3.3). Practically, the map explains why our *post-affine*, backprop-free updates (Section 3.3.4) remain stable even at modest B : for realistic d , post-LN marginals are already close to $\mathcal{N}(0, 1)$, so gentle mean/scale nudges (Equation (3.27) suffice to recenter and rescale) the operating point without altering the learned token-wise projection. Notably, it is observable that domain alignment of LN is more sensitive to the d dimension than batch data B .

3.4.1.2 TTA Evaluation

During our evaluation, results for other methods are from (Mirza *et al.*, 2023), if not otherwise indicated.

ScanObjectNN-C (Table 3.1). Across 16 corruption types, our backpropagation-free, source-free LN-TTA sets a new bar on ScanObjectNN-C with the highest mean accuracy (49.31%, +12.35 over Source Only) while retaining high throughput (64.24 PPS); see Table 3.1. Notably,

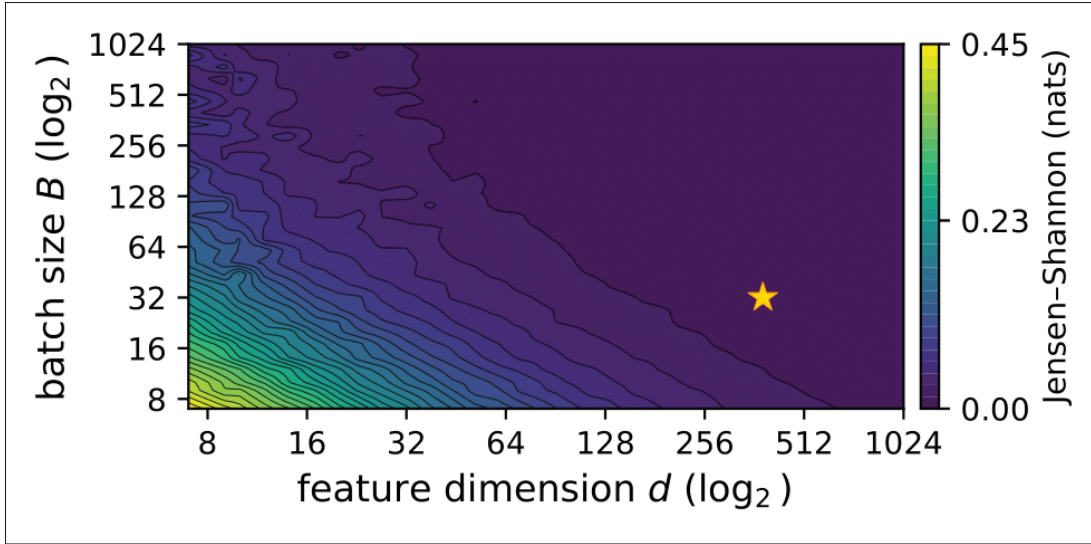


Figure 3.4 Jensen-Shannon (JS) distance between the *average per-feature* post-LN batch marginal and $\mathcal{N}(0, 1)$ over a (B, d) grid. Shown as a contour heat map with $\log_2$ axes. The star marks the (B, d) setting used in our TTA evaluations

LN-TTA outperforms strong *backprop-based* TTA baselines such as TENT (43.45%) and T3A (42.46%) *without* any gradients or source data. The gains are pronounced on structure-altering shifts: e.g., background (25.13 vs. 14.46), impulse noise (40.28 vs. 29.60), RBF and RBF-Inv (56.63/60.59 vs. 50.43/51.81), density increase (74.01 vs. 68.85), shear (56.28 vs. 53.70), rotation (50.95 vs. 44.23), and cut (77.63 vs. 71.43), all compared to TENT. Occlusion and LiDAR remain uniformly challenging across methods (7.40 and 9.81), but the overall mean still clearly favors our approach. Compared to prior *backprop-free* methods, we surpass BFTT3D (33.00%, 1.67 PPS) and DUA (38.66%) by large margins, demonstrating that a post-LN recoloring calibrated once suffices to recover source-regime behavior at test time.

ModelNet-C (Table 3.2). On ModelNet-C, LN-TTA delivers the largest average improvement among *source-free* methods: 69.74% mean accuracy (+15.58 vs. Source Only), exceeding backprop-based TENT (63.53%) and backprop-free BFTT3D (57.16%) and DUA (54.73); see Table 3.2. The method is consistently strong under non-Euclidean, token-irregular corruptions, e.g., density decrease/increase (80.88/83.47), RBF/RBF-Inv noise (76.01/76.42), and shear/rotation (78.32/60.17), and it substantially improves the difficult *occlusion* and *lidar*

cases (52.47/51.42), outperforming all listed alternatives. Crucially, these gains come with the highest throughput among competitive methods (93.03 PPS) and zero test-time optimization, validating that calibrating only the two post-LN degrees of freedom (global scale and common mode) is both effective and deployment-friendly.

ShapeNet-C (Table 3.3). ShapeNet-C further confirms the trend: LN-TTA attains the best *source-free, backprop-free* mean (64.38%, +3.03 over Source Only), surpassing BFTT3D (60.73%) and DUA (60.77%) while matching or beating backprop-based TTA methods on most corruptions; see Table 3.3. The method excels on geometry- and density-driven shifts, e.g., impulse (68.72), upsampling (79.28), RBF/RBF-Inv (77.95/78.20), shear (79.43), rotation (65.25), and cut (83.91), and maintains strong throughput (73.12 PPS). Although occlusion and lidar are particularly hard on this benchmark (DUA edges us out with 11.90/12.10 vs. 9.11/9.41), the aggregate accuracy remains state-of-the-art under the *pure TTA + backprop-free* constraint. Finally, our identical numbers in “Standard” and “Online” regimes highlight a key practical advantage: LN-TTA is stateless and does not rely on streaming gradient updates, simplifying deployment without sacrificing robustness.

3.5 Conclusion

We revisited Layer Normalization (LN) from a batch-domain view: its pre-affine step cancels per-sample shift/scale and concentrates per-feature batch marginals near (0, 1), making cross-batch re-standardization unnecessary. This yields a simple rule: keep the pre-affine LN intact and adjust only the post-affine mean and gain. Instantiated as **LN-TTA**, our backprop- and source-free, stateless TTA applies a single forward pass to uniformly reparameterize LN layers, yielding identical episodic/online predictions, strong gains on three point-cloud corruption suites, and high throughput with minimal memory. **Limitations and future work.** Performance still depends on batch size; reducing this sensitivity and extending to other modalities/backbones are priorities. We will release the implementation to facilitate adoption and further study.

CHAPTER 4

PRISMA: PROMPT REFINEMENT FOR INSTANCE SEGMENTATION OF MODEL-BASED UNSEEN OBJECTS WITH ADAPTATION-FREE SELF-GROUNDING

Moslem Yazdanpanah* , Ali Bahri , Mehrdad Noori , Sahar Dastani , Ismail Ben Ayed ,
Christian Desrosiers

Department of Software Engineering and IT, École de Technologie Supérieure,
1100 Notre-Dame West, Montreal, Quebec, Canada H3C 1K3

Paper submitted to *Pattern Recognition*, December 2025

Abstract

Model-based instance segmentation of unseen objects is critical for tasks such as 6DoF pose estimation and robotic manipulation. Existing training-free pipelines either rely on exhaustive full-scene prompting, which is computationally inefficient, or on language-based grounding, which fails under the modality mismatch problem when industrial objects cannot be distinguished by text. We propose a training- and adaptation-free framework that uses the object itself as the grounding signal. Model-Based Prompt Generation (MPG) condenses multi-view embeddings into prototypes and restricts SAM queries to object-like regions, while Self-Grounding (SG) injects these prototypes into backbone attention layers to separate foreground and background tokens without retraining. A lightweight Confidence-Weighted (CW) scoring further stabilizes predictions. On the BOP benchmark, our method improves mean Average Precision by 1.2 points and reduces runtime by threefold compared to the last SOTA, with gains also propagating to 6DoF pose estimation. This establishes a new Pareto frontier for efficient, deployment-ready segmentation.

4.1 Introduction

Model-based instance segmentation is a key component in many industrial and robotic applications, serving as a critical step for tasks like model-based Six Degrees of Freedom (6DoF)

object pose estimation (Lin *et al.*, 2024; Chen *et al.*, 2024b; Wen *et al.*, 2024; Xiang *et al.*, 2017; Örnek *et al.*, 2025). Accurate pose estimation requires first identifying all instances of an object in the scene and determining their precise region. The performance of industrial tasks, such as robotic grasping and picking, heavily relies on the quality of this segmentation. With advancements in language modeling, language-based prompting (Zhu & Chen, 2024) has emerged as a solution for various tasks, including object segmentation. However, language descriptions often fail to capture the details necessary to distinguish complex industrial objects.

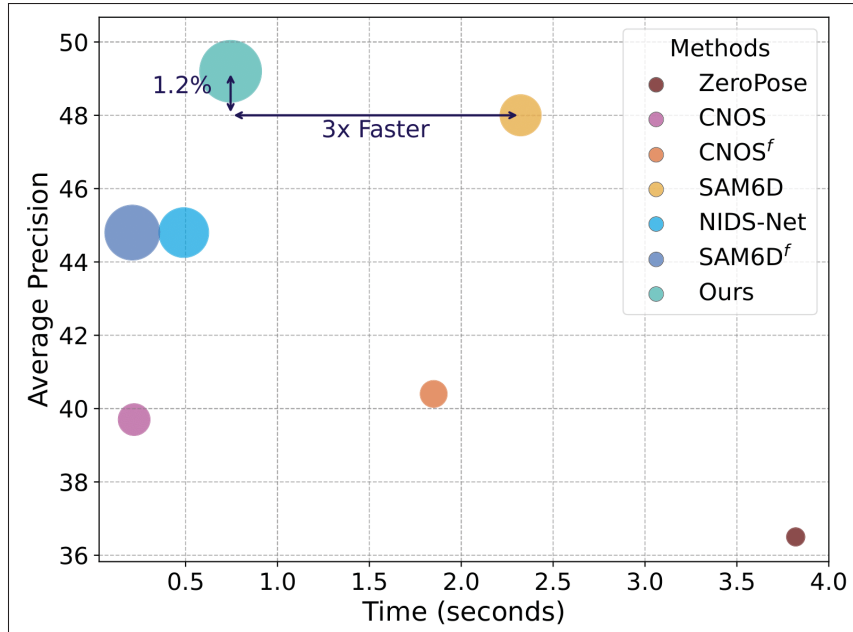


Figure 4.1 Average Precision (AP) and inference Time of our method compared to other SOTA methods in Model-Based Instance Segmentation of Unseen Objects. The bubble size represents the efficiency (AP penalized by time)

Often, industrial objects are absent from public vocabularies, making it challenging for language models to recognize them (see Figure 4.3). We refer to this challenge as the Modality Mismatch Problem (MMP), where all details in one modality (image) cannot be effectively described in another (text). Model-based approaches provide accurate solutions for the MMP by leveraging detailed object models, such as CAD files or multi-view images, to segment all instances of a target object. These approaches excel in accuracy but face a critical limitation: they are typically

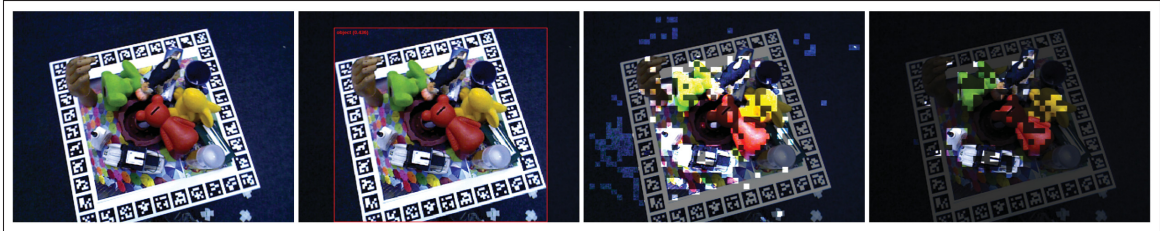


Figure 4.2 Comparison of different prompting strategies. Light areas indicate the prompted regions. From left to right: full scene prompting Lin *et al.* (2024), Grounding-Dino Zhang *et al.* (2022a), Our Model-Based Prompt Generation (MPG), and Our Self-Grounding approach. Grounding-Dino struggles to distinguish individual objects, resulting in inaccurate prompts and insufficient segmentation masks (Maximum of 3 masks). Our MPG module separates the foreground from the background but includes some false positives, while our Self-Grounding module focuses more accurately on foreground areas, significantly reducing false positives. Scene is taken from the HB dataset (Kaskman *et al.*, 2019)

trained on a limited set of objects, hindering their ability to generalize to unseen objects, which is often a requirement for industrial applications.

To address this limitation, the Model-Based Instance Segmentation of Unseen (MISUO) benchmark (Hodan *et al.*, 2024) was introduced. MISUO methods (Chen *et al.*, 2024b; Nguyen *et al.*, 2023; Lin *et al.*, 2024; Lu *et al.*, 2024) aim to perform instance segmentation on arbitrary objects that are novel and unseen during training. While MISUO methods achieve industrial-level performance, they suffer from high computational overhead due to their exhaustive scene-processing approach. Alternative strategies, such as using bounding box prompts generated by Grounding-DINO (Liu *et al.*, 2023), attempt to improve efficiency but struggle with the MMP, failing to accurately detect industrial objects that lack distinct textual descriptions Figure 4.2. A deeper analysis of MMP in Grounding-DINO is provided in Appendix III.

To address these limitations, we propose Model-Based Prompt Generation (MPG), an efficient prompt generation approach for MISUO that is training-free and does not require additional pre-trained models. Our MPG module significantly reduces computational overhead, avoiding exhaustive utilization of SAM during full-scene prompting in the current MISUO methods.

This is achieved by leveraging the embeddings from DINOv2 (Oquab *et al.*, 2023) to identify object-relevant regions in the scene.

Although our MPG approach efficiently focuses the prompts on the object regions, it remains noisy due to background regions having embeddings similar to the target objects (see Figure 4.2). To alleviate this issue, we introduce Self-Grounding, *an intra-modality grounding approach* that uses object-model embeddings to enhance the segmentation of target regions, as illustrated in Figure 4.2. Unlike traditional grounding, Self-Grounding operates entirely within the image modality, thereby addressing the MMP. We implement this method in a training-free manner by modifying the attention layers of DINOv2, allowing for even more efficient and accurate prompt generation.

Our contributions can be summarized as follows:

- We propose a novel integration of model-based prompt generation Model-Based Prompt Generation (MPG), to MISUO methods. Our method, MPG, leverages RGB information from object models and provides an efficient base for MISUO methods, and reduces time complexity in segmentation and scoring.
- We introduce Self-Grounding (SG), an intra-modality grounding approach that bridges example-based concepts within the same modality, with a training-free implementation for ViT architectures;
- We provide extensive analysis of our modules, achieving a threefold speed improvement and a new State Of The Art (SOTA) in accuracy for the MISUO task.

4.2 Related Works

Recent advancements in computer vision have aimed to develop instance segmentation methods that accurately identify and delineate individual objects, which is essential for applications like automated manufacturing, quality inspection, and autonomous robotic manipulation. Early methods such as Faster R-CNN (Girshick, 2015) and Mask R-CNN (He *et al.*, 2017) established the two-stage detection–segmentation paradigm and provided robust results when training data

is abundant and object categories are fixed. However, these methods typically fail to generalize beyond their labeled categories, limiting their applicability in open-world or industrial scenarios where new objects are constantly introduced. To address such generalization challenges, ShapeMask (Kuo *et al.*, 2019) and NOCS (Wang *et al.*, 2019a) attempted to incorporate category-level priors, enabling recognition of unseen objects within broad categories. Nevertheless, these approaches remain constrained by category definitions and struggle when faced with objects that do not conform to existing specifications, such as textureless industrial parts or custom components.

Model-Based Instance Segmentation of Unseen (MISUO) The introduction of benchmarks such as BOP (Hodan *et al.*, 2024), originally motivated by 6DoF object pose estimation, has spurred research on the more challenging MISUO task. Unlike category-level approaches, MISUO methods are object-specific but training-free: they must segment instances of novel objects using only a CAD model or a small collection of multi-view images as reference. This problem formulation captures the requirements of real-world robotics, where training for every new object is infeasible, yet precise segmentation remains mandatory.

Among recent works, ZeroPose (Chen *et al.*, 2024b) leverages SAM (Kirillov *et al.*, 2023) to densely propose segmentations from the entire scene, and then selects relevant masks by comparing their embeddings with those of the reference object. CNOS (Nguyen *et al.*, 2023) follows a similar philosophy: SAM is used to generate proposals, which are then filtered using feature similarity against 3D templates of the object’s category. CNOS further removes proposals that are inconsistent with geometric or semantic properties of the category, making it robust across a variety of novel objects. SAM6D (Lin *et al.*, 2024) pushes this line of work further by integrating SAM for segmentation and refining proposals using multiple similarity cues, appearance, semantics, and geometry, before selecting high-confidence masks. While effective, these methods inherit the computational cost of full-scene prompting, since they query SAM exhaustively and discard background proposals afterward. This bottleneck is one of the major limitations that motivates our work.



Figure 4.3 Examples of similar industrial objects from the T-LESS dataset (Hodan *et al.*, 2017) (objects 13–16). Due to the MMP, these objects are indistinguishable using language-based descriptors, underscoring the limitations of text-driven prompting

Prompting and Object Detection for Segmentation An alternative line of work explores prompt-based segmentation, where external cues such as bounding boxes or textual descriptions restrict the search space of SAM. Grounding-DINO (Liu *et al.*, 2023) exemplifies this idea by employing open-set object detection through cross-modal grounding. It maps text queries into bounding box prompts via a language encoder (BERT) and then uses SAM to segment inside those regions. This strategy reduces the number of proposals compared to full-scene prompting. However, it suffers from the MMP: many industrial objects are nearly indistinguishable in language, and textual cues fail to disambiguate them. Grounded SAM (Ren *et al.*, 2024) attempts to address this by integrating Grounding-DINO with SAM for text-driven segmentation, but ultimately remains vulnerable to the same modality mismatch.

Positioning Our Work Our approach situates itself at the intersection of these two paradigms. Like SAM6D and CNOS, we aim for training-free MISUO, but unlike them we avoid the heavy cost of full-scene prompting. Like prompt-based methods, we reduce the number of SAM queries, but we do so without relying on textual grounding and hence avoid the pitfalls of the MMP. By combining MPG, which performs prototype-guided sparsification of prompts, and Self-Grounding (SG), which performs object-conditioned token routing inside the backbone, we introduce a solution that is both efficient and robust. Together with Confidence-Weighted Scoring (CW), which stabilizes predictions using the segmenter’s own reliability signal, our method offers a balanced path forward: it preserves the precision of model-based approaches

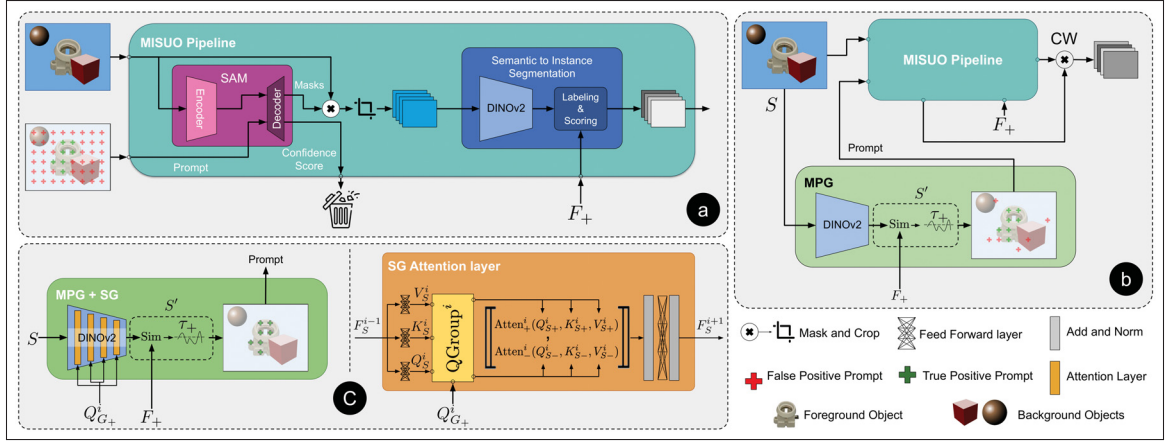


Figure 4.4 The notations in this figure follow Section 4.3. **a)** MISUO pipeline (e.g., SAM6D): SAM receives full-scene prompts to produce candidate masks, which are then labeled and scored (Lin *et al.*, 2024). SAM’s internal confidence scores are not fully exploited. **b)** Our MPG module reuses existing backbones and generates a sparse set of foreground prompts for SAM, reducing redundant queries. **c)** Left: our Self-Grounding (SG) module injects object embeddings into the scene backbone via intra-modality grounding, separating foreground from background and reducing false-positive prompts; Right: SG attention layer details. Skip connections are omitted for clarity. The operator $\llbracket \cdot, \cdot \rrbracket$ denotes position-aware concatenation

while delivering the efficiency of prompt-based ones. This positions our framework as a practical step toward deploying training-free segmentation in real-world industrial applications.

4.3 Method

4.3.1 Task Definition

Model-Based Instance Segmentation of Unseen Objects (MISUO) aims to segment novel objects for which we have either CAD models or multi-view images, but no large-scale training data for those specific instances. At inference, the goal is to produce segmentation masks for each instance of the target objects, together with confidence scores. Although some setups may provide a training set containing *different* objects or scenes, we focus on the *training-free* MISUO setting, employing only off-the-shelf pretrained models: SAM (Kirillov *et al.*, 2023) for segmentation and DINOv2 for mask filtering and scoring (Fig. 4.4-a).

4.3.2 Preliminaries

4.3.2.1 Object-Model Representation

We assume access to either 3D CAD models or multi-view images for the objects of interest:

$$\mathbb{O} = \{O_1, \dots, O_{n_O}\}. \quad (4.1)$$

In practice, capturing multi-view images is often easier than obtaining accurate CAD models. Let

$$\mathbb{C} = \{C_1, \dots, C_{n_C}\} \quad (4.2)$$

denote the camera views around an object O_o . We denote the resulting multi-view image set by $\mathbb{M}_C^{O_o}$. To obtain object descriptors, we feed $\mathbb{M}_C^{O_o}$ through a pretrained DINOv2 backbone and cache:

- a *local* tensor of embeddings

$$\mathbf{F}_O^\ell \in \mathbb{R}^{n_O \times n_C \times H'W' \times d}, \quad (4.3)$$

containing one d -dimensional token per spatial patch per view;

- a *global* tensor of embeddings

$$\mathbf{F}_O^g \in \mathbb{R}^{n_O \times n_C \times d}, \quad (4.4)$$

containing one d -dimensional global token per view.

Here, $H'W'$ is the number of spatial patches per view and d is the feature dimension. By caching $\mathbf{F}_O^\ell$ and $\mathbf{F}_O^g$ offline, we avoid recomputing object features at inference.

Cosine Similarity. Throughout, we measure similarity between vectors $\mathbf{f}_1, \mathbf{f}_2 \in \mathbb{R}^d$ by

$$\text{Sim}(\mathbf{f}_1, \mathbf{f}_2) = \frac{\mathbf{f}_1^\top \mathbf{f}_2}{\|\mathbf{f}_1\|_2 \|\mathbf{f}_2\|_2}. \quad (4.5)$$

4.3.3 Model-Based Prompt Generation (MPG)

Many MISUO methods generate *full-scene prompts* for SAM: they query every spatial location and then filter proposals downstream (Lin *et al.*, 2024). While exhaustive, this is inefficient in industrial settings:

- it produces $\mathcal{O}(N_S)$ proposals for N_S patches, many of which are redundant or clearly background;
- it complicates scoring, as noisy background masks must be aggressively pruned.

Full-scene prompting also stresses memory and runtime, which is undesirable in real-time or high-throughput use.

Instead, MPG restricts prompts to patches that are likely to belong to the target objects. Let $\mathbf{f}_S^\ell[p] \in \mathbb{R}^d$ denote the DINOv2 embedding of the p -th patch of a test scene S , and let $\mathbf{f}_+ \in \mathbb{R}^d$ be an “object-like” prototype. If we keep only patches with

$$\text{Sim}(\mathbf{f}_+, \mathbf{f}_S^\ell[p]) > \tau_+, \quad (4.6)$$

we retain approximately αN_S patches, where $\alpha \ll 1$ if objects occupy a small fraction of the scene.

We construct the prototype $\mathbf{f}_+$ by averaging over local object embeddings:

$$\mathbf{f}_+ = \frac{1}{n_O n_C H' W'} \sum_{o=1}^{n_O} \sum_{c=1}^{n_C} \sum_{hw=1}^{H' W'} \mathbf{F}_O^\ell(o, c, hw, :) \in \mathbb{R}^d. \quad (4.7)$$

For a test scene S , we compute its patch-level embedding matrix

$$\mathbf{F}_S^\ell \in \mathbb{R}^{H' W' \times d}, \quad (4.8)$$

whose rows are patch descriptors. We then build a binary mask $\mathbf{m}_S \in \{0, 1\}^{H' W'}$ by thresholding

$$\mathbf{m}_S[p] = [\text{Sim}(\mathbf{f}_+, \mathbf{F}_S^\ell[p, :]) > \tau_+], \quad (4.9)$$

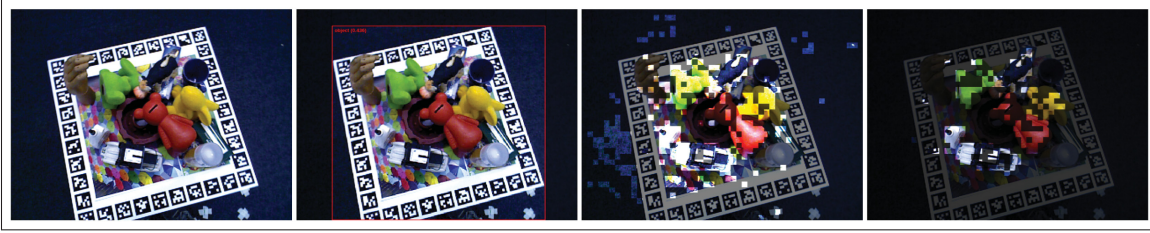


Figure 4.5 Prompting strategies for MISUO. **Left:** full-scene prompting (Lin *et al.*, 2024) queries every region, producing exhaustive but inefficient proposals. **Second:** text-based Grounding-DINO (Zhang *et al.*, 2022a) often fails in multi-model product (MMP) settings, merging or missing objects. **Third:** our MPG focuses prompts on object-like regions, reducing redundancy but leaving some false positives. **Right:** Self-Grounding (SG) further refines prompting by re-routing attention in the backbone, sharply focusing on true foreground areas. Example from HB (Kaskman *et al.*, 2019)

marking patches that appear *object-like*. Only these patches are used to form prompts for SAM, reducing the query set from $O(N_S)$ to $O(\alpha N_S)$.

Thus MPG: (1) reduces segmentation overhead by focusing SAM on far fewer prompts, and (2) improves the proposal pool by emphasizing object regions and suppressing obvious background clutter. Fig. 4.5 illustrates the difference between full-scene prompting, text-based grounding, and our MPG/SG pipeline.

4.3.4 Self-Grounding

Although MPG concentrates prompts on object-like patches, it may still admit false positives, especially when background structures resemble the target. Text-based grounding can sometimes help, but fails under MMP conditions where objects lack distinctive textual labels. We therefore propose a *same-modality* grounding mechanism, *Self-Grounding* (SG), that refines scene embeddings by referencing stored object embeddings in the *same* DINOv2 feature space.

After embedding the object multi-views $\mathbb{M}_C^O$ through DINOv2, we collect *layer-based* grounding data for transformer layers $i = 1, \dots, L$:

$$G^i = (\mathbf{Q}_O^i, \mathbf{K}_O^i, \mathbf{V}_O^i), \quad (4.10)$$

where $\mathbf{Q}_O^i, \mathbf{K}_O^i, \mathbf{V}_O^i$ stack tokens over all objects O_o , cameras C_c and spatial patches hw . We aggregate over $n_O, n_C, H'W'$ to form a positive grounding triple

$$G_+^i = (\mathbf{Q}_{G_+}^i, \mathbf{K}_{G_+}^i, \mathbf{V}_{G_+}^i), \quad (4.11)$$

where, for example,

$$\mathbf{Q}_{G_+}^i = \frac{1}{n_O n_C H'W'} \sum_{o=1}^{n_O} \sum_{c=1}^{n_C} \sum_{hw=1}^{H'W'} \mathbf{Q}_O^i(o, c, hw, :), \quad (4.12)$$

and similarly for $\mathbf{K}_{G_+}^i, \mathbf{V}_{G_+}^i$.

When processing a scene S at layer i , the backbone computes

$$(\mathbf{Q}_S^i, \mathbf{K}_S^i, \mathbf{V}_S^i), \quad (4.13)$$

with one row per patch. We partition patches into *foreground* and *background* via a similarity-based mask

$$\mathbf{m}_{\text{fg}}^i[p] = [\text{Sim}(\mathbf{Q}_{G_+}^i, \mathbf{Q}_S^i[p, :]) > \tau_G], \quad (4.14)$$

where τ_G is a global threshold. Patches with $\mathbf{m}_{\text{fg}}^i[p] = 1$ form the foreground group; the others form the background group. We then apply standard ViT attention *within* each group:

$$\mathbf{A}_+^i = \text{Atten}(\mathbf{Q}_{S_+}^i, \mathbf{K}_{S_+}^i, \mathbf{V}_{S_+}^i), \quad \mathbf{A}_-^i = \text{Atten}(\mathbf{Q}_{S_-}^i, \mathbf{K}_{S_-}^i, \mathbf{V}_{S_-}^i), \quad (4.15)$$

where the subscripts $+$ and $-$ denote the subsets of rows selected by $\mathbf{m}_{\text{fg}}^i$ and its complement, respectively. Finally, we merge the outputs by position-aware concatenation:

$$\text{SGAtten} = \llbracket \mathbf{A}_+^i, \mathbf{A}_-^i \rrbracket. \quad (4.16)$$

This grouping “gates” the scene tokens so that patches matching $\mathbf{Q}_{G_+}^i$ are treated as foreground in $\mathbf{A}_+^i$. No training is needed: we merely reorder tokens based on similarity to $\mathbf{Q}_{G_+}^i$, then apply

standard attention. Repeating this at selected layers drives attention toward object regions and suppresses distractors, complementing MPG’s prompt-level filtering.

4.3.5 Confidence-Weighted IoU Scoring (CW)

To improve the scoring step in existing MISUO pipelines, we incorporate SAM’s mask stability scores into the final ranking. Given N masks $\{\mathbf{M}_i\}_{i=1}^N$ from SAM and their IoU scores $\{q_i\}$, we first compute a *stability score*

$$s_i = \frac{|\mathbf{M}_i^{\tau_I+\delta} \cap \mathbf{M}_i^{\tau_I-\delta}|}{|\mathbf{M}_i^{\tau_I+\delta} \cup \mathbf{M}_i^{\tau_I-\delta}|}, \quad \mathbf{M}_i^{\tau_I \pm \delta} = [\mathbf{M}_i > \tau_I \pm \delta], \quad (4.17)$$

where τ_I is SAM’s default IoU threshold and $\delta > 0$ is a small offset. We then keep only masks whose stability exceeds a user-chosen cutoff τ_s :

$$\text{CW} = \{q_i : s_i \geq \tau_s\}. \quad (4.18)$$

This confidence-weighted filtering discards unstable masks while preserving reliable, high-IoU candidates. In our full MISUO pipeline, this CW term is multiplied with the semantic/appearance/geometric matching score SC_m (defined in the Appendix) to produce a final confidence score per instance.

4.4 Experiments

In this section, we evaluate our proposed method (MPG, SG, CW) on three challenging tasks: Model-Based Instance Segmentation of Unseen (MISUO), model-based detection, and 6DoF pose estimation of unseen objects, using the BOP benchmark (Hodan *et al.*, 2024). This benchmark includes seven datasets: LM-O (Brachmann *et al.*, 2014), T-LESS (Hodan *et al.*, 2017), ITODD (Drost *et al.*, 2017), HB (Kaskman *et al.*, 2019), YCB-V (Xiang *et al.*, 2017), TUD-L (Hodan *et al.*, 2024), and IC-BIN (Doumanoglou, Kouskouridas, Malassiotis & Kim, 2016), covering diverse conditions such as lighting variations, texture-less objects, and occlusions. Segmentation and detection accuracy (Table 4.1, Table 4.3) follow COCO 2020 methodology (Lin

et al., 2014), using Average Precision (AP) across IoU thresholds [0.5 to 0.95]. Per-object AP (APO) averages into Average Precision over Dataset (APD) and is further averaged as “Mean” accuracy. Only instances with at least 10% visible area and up to 100 predictions per image are evaluated. Pose estimation accuracy uses VSD, MSSD, and MSPD metrics. A pose is correct if its error is below a threshold, with Average Recall (AR_e) calculated across thresholds. Dataset accuracy (ARD) averages ARVSD, ARMSSD, and ARMSPD, with overall accuracy reported as “Mean” (Table 4.2). Further details on datasets and metrics can be found in (Hodan *et al.*, 2024).

Implementation Details. For implementing our method, we used the SAM6D (Lin *et al.*, 2024) code base and followed the exact setting and configuration from this method. In short, we used the huge model for Segment Anything Model (SAM), and the large model for DINOv2. Moreover, we set the similarity threshold of the MPG module τ_+ to 0.6 across all datasets. For the scene-based foreground threshold τ_G of the SG module, we used a value of 0.7 as about 30% of the scenes in our dataset are foreground, while the rest is background. We also tested threshold values of 0.6 and 0.8, and observed a slight drop in performance (less than 0.1%). In this study, we employed a fixed value of τ_G for all layers and all datasets; however, one could achieve a better runtime and accuracy by tuning these hyperparameters for each dataset. Similarly, we have used $\tau_s = 0.85$ across all the datasets and evaluations. To avoid redundancy, please refer to (Lin *et al.*, 2024) for more details on other settings. All the experiments are conducted on an A600 GPU. To ensure a fair comparison, we have re-run our baseline (SAM6D) on the same machine.

4.4.1 Model-Based Instance Segmentation of Unseen (MISUO)

As can be seen from Table 4.1, our method distinguishes itself on the BOP benchmark for the MISUO task by significantly reducing computation time while maintaining high accuracy, all without utilizing additional backbones beyond DINOv2. Specifically, our approach is approximately three times faster than SAM6D, cutting down the run time per instance from SAM6D’s 2.32 seconds to just 0.77 seconds. This efficiency is achieved without sacrificing performance or relying on extra computational resources. In terms of dataset-specific

improvements, our method consistently outperforms SAM6D across several datasets. For instance, we achieve accuracy improvements of up to 1-2% on datasets like LM-O, T-LESS, TUD-L, IC-BIN, ITODD, and YCB-V with an average improvement of 1.2% over all datasets. Importantly, our method does not require test-time adaptation, unlike methods such as NIDS-Ne-CA and NIDS-Net-WA, which introduce additional complexity and extra adaptation times. By integrating techniques like SG and CW into our base method (MPG), we enhance both efficiency and accuracy without the need for multiple backbones or increased computation time. A comparison of prompt quality between our method and SAM6D is presented in Appendix III-6.2.4.

Table 4.1 Evaluation of our method in BOP benchmark (Hodan *et al.*, 2024), the Model-Based Instance Segmentation of Unseen (MISUO) task. Methods with superscript ^f use FastSAM for segmentation, and the rest use SAM. In our method, SG stands for Self-Grounding and CW stands for confidence weighting. Results for methods with ● (baseline) are reproduced on our hardware to have a fair time comparison. ● showcases the methods with test-time adaptation. The time for NIDS-Net + CA, and NIDS-Net + WA is unknown due to the unknown adaptation time. GD in Extra Backbones consists of Clip (Radford *et al.*, 2021), and DINO (Zhang *et al.*, 2022a). Reported numbers are Average Precision

BOP Datasets										
Method	Extra	LM-O	T-LESS	TUD-L	IC-BIN	ITODD	HB	YCB-V	Mean	Time
Methods with adaptation (Backbone Update)										
NIDS-Net pure	GD, DINOv2	42.9	43.0	52.0	30.5	28.8	56.6	59.7	44.8	0.49
NIDS-Net CA 		42.8	47.5	50.0	29.7	27.6	62.7	63.5	46.3	–
NIDS-Net WA 		43.9	49.6	55.6	32.8	31.5	62.0	65.0	48.6	–
Train-Free methods without adaptation										
ZeroPose	DINOv2	34.4	32.7	41.4	25.1	22.4	47.8	51.9	36.5	3.82
CNOS ^f		39.7	37.4	48.0	27.0	25.4	51.1	59.9	41.2	0.22
CNOS		39.6	39.7	39.1	28.4	28.2	48.0	59.5	40.4	1.85
SAM6D ^f 		42.4	41.6	51.7	30.1	31.4	54.3	61.9	44.8	0.21
SAM6D 		46.3	44.3	57.5	36.2	32.1	59.1	60.4	48.0	2.32
Ours-MPG		46.0	45.8	58.7	36.9	33.4	57.9	60.7	48.5	1.43
Ours-MPG+SG	DINOv2	46.2	45.4	59.0	37.0	33.6	57.0	60.7	48.4	0.77
Ours-MPG+SG+CW		47.1	46.2	58.6	37.9	35.2	57.6	61.8	49.2	0.77

4.4.2 6DOF Pose Estimation of Unseen Objects

Since accurate 6DoF pose estimation heavily depends on the quality of segmentation, we have evaluated how our method affects downstream pose estimation of unseen objects. To emphasize efficiency, we have categorized approaches into fast methods (runtime less than 3

seconds) and slow methods. From Table 4.2, our method (our segmentation + SAM6D pose estimation) establishes a new SOTA performance among fast methods by achieving the highest mean of Average Recalls, keeping the runtime of just under 2 seconds per scene. Compared to other fast methods, we demonstrate significant improvements. For instance, we outperform SAM6D^f, the next best fast method with a notable increase in mean AR (3.6%), despite a comparable runtime. Dataset-specific results further highlight our method’s superiority: on datasets like LM-O, T-LESS, TUD-L, IC-BIN, and ITODD, we achieve accuracy improvements over SAM6D^f ranging from approximately 1% to over 13%. For example, on LM-O and TUD-L, we improve accuracy by about 4% and 8% respectively, while on IC-BIN, the improvement is around 13%. While we are slightly behind SAM6D^f on a few datasets by a small margin, our overall performance demonstrates robustness and consistency. Notably, our method approaches the accuracy of slow methods like SAM6D, which operates with a significantly longer runtime, maintaining high performance without the increased computational burden.

Table 4.2 Evaluation of the effect of our method on downstream 6DoF pose estimation of unseen objects. Methods with superscript ^f use FastSAM for segmentation. In FoundationPose (Wen *et al.*, 2024), GT showcases corrupted ground-truth segmentation masks. In FoundPose (Örnek *et al.*, 2025), letters C and F represent Coarse and Feature-refinement respectively. Results for methods with ● are reproduced on our hardware to have a fair time comparison. Reported numbers are Average Recall

Method		BOP Datasets							Mean	Time
Segmentation	Pose Estimation	LM-O	T-LESS	TUD-L	IC-BIN	ITODD	HB	YCB-V		
Slow Methods										
CNOS	GenFlow	63.5	52.1	86.2	53.4	55.4	77.9	83.3	67.4	34.6
SAM6D	FreeZe	67.6	50.0	88.1	48.7	52.0	76.1	77.4	65.7	11.47
GT	FoundationPose	75.6	64.6	92.3	50.8	58.0	83.5	88.9	73.4	29.32
SAM6D 	SAM6D	70.9	54.0	91.3	56.6	56.0	72.1	80.8	68.8	4.36
Fast Methods										
CNOS	GigaPose	29.6	26.4	30.0	22.3	17.5	34.1	27.8	26.8	0.38
CNOS	FoundPose-C	39.7	33.8	46.9	23.9	20.4	50.8	45.2	37.3	1.69
CNOS	FoundPose-F	39.5	39.6	56.7	28.3	26.2	58.5	49.7	42.6	2.64
SAM6D ^f 	SAM6D	68.2	51.0	85.5	44.0	54.6	69.4	80.0	64.7	1.45
Ours	SAM6D	72.4	52.0	93.2	57.4	56.8	68.9	77.2	68.3	1.92

4.4.3 Model-Based 2D Detection of Unseen Objects

The accurate detection of unseen objects ensures that subsequent processes operate on reliable data with precise regions of the target object. As reported in Table 4.3, our method excels in this

critical task by achieving higher accuracy across multiple datasets without the need for additional backbones or test-time adaptation. Specifically, we outperform SAM6D with consistent accuracy improvements ranging from 1% to 3% on datasets such as LM-O, T-LESS, TUD-L, IC-BIN, and ITODD, showcasing our method’s robustness and versatility. The integration of SG and CW into our base method (MPG) enhances detection accuracy by enabling better localization and more reliable confidence assessments, without increasing computational complexity. Importantly, the computation time for detection remains the same as for segmentation in our approach, emphasizing efficiency without compromising performance.

Table 4.3 Evaluation of our method in BOP benchmark (Hodan *et al.*, 2024), the task of **Model-Based 2D Detection of Unseen Objects**. The segmentation model is SAM for all methods. In our method, SG stands for Self-Grounding and CW stands for confidence weighting. ● showcase the methods with test-time adaptation. The time for NIDS-Net + WA is unknown due to the unknown adaptation time. GD in Extra Backbones consists of Clip (Radford *et al.*, 2021), and DINO (Zhang *et al.*, 2022a). Reported numbers are Average Precision

BOP Datasets										
Method	Extra	LM-O	T-LESS	TUD-L	IC-BIN	ITODD	HB	YCB-V	Mean	Time
Methods with adaptation (Backbone Update)										
NIDS-Net pure	GD ,	44.9	42.8	43.4	24.4	34.9	54.8	56.5	43.1	0.49
NIDS-Net WA ●	DINOv2	45.7	49.3	48.6	25.7	37.9	58.7	62.1	46.9	–
Train-Free methods without adaptation										
ZeroPose		36.7	30.0	43.1	22.8	25.0	39.8	41.6	34.1	3.82
CNOS	DINOv2	39.5	33.0	36.8	20.7	31.3	42.3	49.0	36.1	1.85
SAM6D		46.6	43.7	53.7	26.1	39.3	53.1	51.9	44.9	2.32
Ours-MPG		46.2	43.7	57.0	26.2	40.0	51.1	52.9	45.3	1.43
Ours-MPG+SG	DINOv2	46.5	43.4	57.4	26.6	40.4	51.0	53.4	45.5	0.77
Ours-MPG+SG+CW		47.8	44.7	58.4	28.3	42.9	51.8	55.7	47.1	0.77

4.4.4 Ablation Studies and Analyses

Effect of Similarity Threshold. Figure 4.1 demonstrates the effect of different similarity thresholds τ_+ , on runtime and Average precision of our method on the LM-O dataset. As the similarity threshold increases, the runtime consistently decreases, indicating that higher thresholds reduce the background prompts and hence, computational demand. However, the accuracy remains relatively stable up to a threshold of approximately $\tau_+ = 0.6$, after which it drops sharply. The selected threshold, shown by the broken line at $\tau_+ = 0.6$, represents a

balance between efficiency and accuracy, minimizing runtime while preserving optimal accuracy. Also, the effect of different values of τ_+ on the prompt area is visualized in Appendix III-6.2.1. This threshold choice achieves a practical trade-off between performance and speed in our experiments Table 4.1, Table 4.2, and Table 4.3.

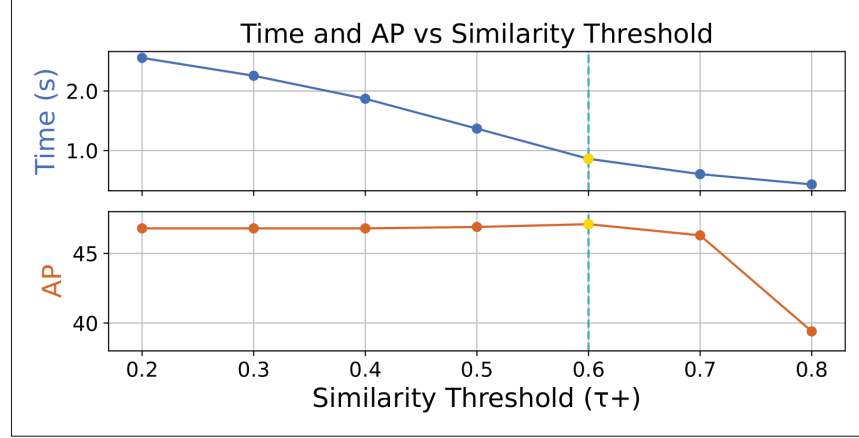


Figure 4.6 Effect of different similarity thresholds τ_+ on run-time (Top) and accuracy (Bottom) of our method over LM-O dataset (Brachmann *et al.*, 2014). The broken line shows the threshold we have used in our experiments

Effect of Proposed Components. The results in Table 4.1, and Table 4.3 illustrate the effect of our proposed components (MPG, MPG+SG, and MPG+SG+CW) compared to the baseline method, SAM6D. Starting with Model-Based Prompt Generation (MPG), it achieved considerable improvements in time performance compared to SAM6D. MPG reduces runtime by around half, making it highly efficient without sacrificing performance. Although we did not expect an improvement of Average Precision (AP) from MPG, we witnessed a 0.5% increase on average for all datasets. This suggests that a higher density of prompts in foreground areas can better detect all possible objects compared to having prompts uniformly distributed across the scene (SAM6D). This reduction in time is crucial for real-world applications where speed is as important as accuracy, positioning MPG as a strong, streamlined alternative to full-scene prompting.

When we add Self-Grounding (SG) in MPG+SG, we observe further time reductions. SG optimizes the model’s ability to identify relevant features to the object’s embeddings, which

translates into less noisy background prompts and faster processing times across datasets. Although this version shows only a modest gain in accuracy (0.4%), the substantial runtime improvement (three times) highlights SG’s effectiveness in enhancing efficiency. Finally, the addition of Confidence-Weighted Scoring (CW) in MPG+SG+CW notably boosts the Average Precision. CW refines predictions by prioritizing areas of high confidence, leading to considerable accuracy improvements, especially in challenging datasets. This configuration achieves the highest accuracy among all variations (0.8% of average improvement), showcasing its ability to deliver precise results while keeping runtime low.

Effect of Confidence-Weighted Scoring (CW). Figure 4.7 summarizes the performance of our base methods with and without the CW module. Notably, the combination of CW with MPG and SG achieved a considerable improvement, while the SAM6D method experienced a slight decrease in performance when combined with CW. This disparity arises because the effectiveness of the CW module heavily depends on the quality of the segmentation masks. Methods like SAM6D, which segment the entire scene, produce masks with substantial background areas. Applying CW to these masks can prioritize background regions over real object masks, leading to a performance drop. In contrast, our MPG and SG components generate more focused segmentation masks (as can be seen from Figure 4.2) with less background interference, allowing the CW module to effectively enhance their performance. This suggests that the CW module is not the sole source of improvement but works synergistically with methods that produce precise segmentation masks.

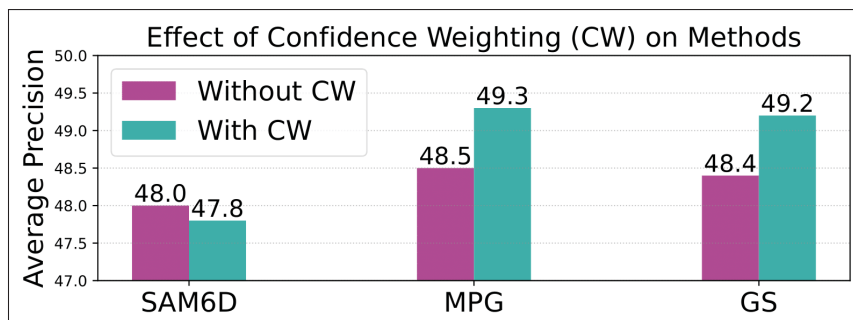


Figure 4.7 Effect of Confidence-Weighted Scoring (CW) on SAM6D and different components of our method, MPG, and SG

Effect of τ_G on performance and efficiency. From Figure 4.8 it is observable that different

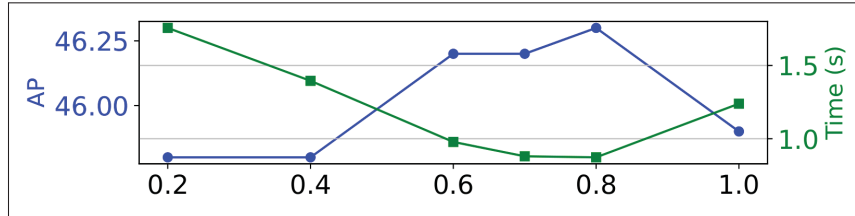


Figure 4.8 Ablation of τ_G on time and accuracy, on LM-O dataset

ratios of positive and negative grouping in SG, can effectively affect the performance and efficiency of the SG component. From 0.2 to 0.8, we witness a steady reduction in the average execution time of the subsequent segmentation, stemming from the increased quality of the generated prompts. Similarly, from 0.4 to 0.8, we observe a continuous increase in the average precision accuracy. From 0.8, the restricted grouping threshold will limit the influence tokens to attend the foreground, resulting in a performance reduction and a decrease in efficiency. During our evaluations, we have set $\tau_G = 0.7$, showcasing that exhaustive parameter tuning can achieve even better results using our algorithm.

For more analysis, and also the details of Table 4.1, Table 4.2, Table 4.3, please refer to Appendix III-6.3.

4.5 Conclusion

We presented a training-free and adaptation-free framework for model-based instance segmentation of unseen objects that combines MPG and SG to operate entirely within the image modality, thereby sidestepping the MMP. MPG reduces the prompt budget by concentrating SAM queries on object-like regions derived from model prototypes, while SG re-routes attention inside the backbone to separate foreground from background tokens, sharpening features without updating weights. Together with lightweight CW at selection time, this leads to a consistent Pareto improvement on BOP: a $\sim 3\times$ speedup over the last SOTA at comparable or better accuracy, culminating in a mean AP gain of +1.2 points and stronger downstream 6DoF pose

and 2D detection performance. Beyond the specific backbone choices, our results suggest that prototype-guided prompt sparsification and groupwise, object-conditioned attention are broadly useful primitives for efficient, deployment-ready MISUO in industrial settings. Future extensions include adaptive routing schedules across layers, backbone-agnostic token partitioners, and integrating geometric priors from CAD to further tighten the coupling between segmentation and pose.

CONCLUSION AND RECOMMENDATIONS

In this thesis, we studied how to bridge the gap between powerful Vision Transformer backbones and the constraints of real 3D and industrial deployments. Rather than relying on heavy retraining or gradient-based Test-Time Adaptation (TTA), we focused on backpropagation-free, forward-only mechanisms that operate directly on normalization layers, token sets, and visual prototypes. Our methods are designed to work with frozen backbones, without access to source data or labels at deployment, and to remain compatible with the tight memory and latency constraints of practical systems.

In Chapter 2, we showed that domain shift in 3D point cloud Transformers can be tackled effectively at the token level. Through the proposed Purge-Gate (PG), we introduced a backpropagation-free TTA mechanism that assigns a shift score to each token and removes the most corrupted ones before they enter the attention layers. This simple idea, instantiated in variants that either use source statistics or are fully source-free via a CLS-based prototype, consistently improves robustness across multiple corrupted and real-world point cloud benchmarks, while also reducing memory usage and inference time. In practice, this demonstrates that carefully designed token selection can be a powerful alternative to iterative gradient-based adaptation.

We then, in Chapter 3, revisited Layer Normalization (LN) from a domain perspective and explained why many BatchNorm-inspired TTA tricks do not transfer to Transformer backbones. We theoretically showed that, as feature dimension and batch size grow, the pre-affine outputs of LN concentrate around zero mean and unit variance, so re-estimating batch statistics is both unnecessary and potentially harmful. Building on this analysis, we proposed a backpropagation-free method that leaves the projection-and-normalization part of LN untouched and adjusts only the affine parameters (gain and bias) in a single forward pass. This lightweight recalibration yields substantial gains on several 3D point cloud benchmarks and provides a clear design rule:

for LN-based architectures, adaptation should act post-affine on the few degrees of freedom that survive normalization, instead of trying to mimic batch-level statistics.

Finally, in Chapter 4, we extended our focus beyond classification to the more demanding task of model-based instance segmentation of unseen objects. We introduced a training-free framework that leverages CAD models or a few reference images to refine prompts for large segmentation models. The method combines three complementary components: model-based prompt generation, which concentrates prompts on object-like regions based on strong visual descriptors; self-grounding, which injects object prototypes into the backbone and enforces a separation between foreground and background tokens; and confidence-weighted scoring, which reuses the segmenter’s own confidence signals to rank predictions. Together, these modules deliver faster and more accurate segmentation of unseen objects and improve downstream 6DoF pose estimation and 2D detection, all without any additional training.

Overall, this thesis shows that robust deployment of modern 2D/3D vision models does not require heavy, gradient-based adaptation loops. By operating directly on normalization parameters, token sets, and visual prototypes, we can design analytical, backpropagation-free mechanisms that are simple to implement, easy to tune, and well suited to real-world constraints. At the same time, our analyses clarify when and why such mechanisms work, especially in the context of Layer Normalization and domain shift in Transformers.

Recommendations. The findings of this thesis lead to several recommendations for both practitioners and researchers:

- *Treat domain shift as the default, not the exception.* Deploying a pretrained model without any adaptation strategy is risky in real 3D and industrial environments. Every model intended for deployment should be accompanied by a lightweight adaptation or steering mechanism, whether built into the backbone (e.g., LN-based recalibration) or provided as an external module (e.g., token-purging or prompting frameworks).

- *Favor forward-only, modular adaptations.* When latency, memory, and data access are constrained, gradient-based TTA or full fine-tuning are often impractical. In such settings, forward-only updates that act on normalization layers, token selection, or prototype injection offer a better trade-off between robustness and engineering complexity.
- *Exploit task-specific structure.* For 3D point clouds, token-level operations that respect geometric structure (such as purging corrupted or low-confidence tokens) are more effective than blindly re-optimizing all parameters. For model-based unseen object segmentation, leveraging CAD models or a few reference views to build visual prototypes and using them to guide prompts and attention, yields significant gains without additional labels.
- *Design adaptation mechanisms to be backbone-aware.* Methods that assume BatchNorm-style behavior can fail on Transformer backbones dominated by Layer Normalization. Future work should continue to develop adaptation techniques that explicitly account for the normalization and architecture used by the model, rather than transferring heuristics from convolutional networks.
- *Push toward domain-universal, backpropagation-free TTA.* As large vision and vision–language foundation models become more prevalent, retraining or test-time training on every new scenario will become increasingly unrealistic. We recommend further research into domain-universal, stateless TTA mechanisms that can be plugged into a wide range of backbones, and into tighter couplings between 3D geometry, prototypes, and attention patterns in segmentation models.

In summary, the results of this thesis suggest that backpropagation-free test-time adaptation is a viable and promising route for making modern vision transformers robust and deployment-ready in challenging 3D and industrial settings. We hope that these ideas will inspire new methods that are even more domain-agnostic, easier to integrate into real systems, and better aligned with the practical constraints of future perception pipelines.

APPENDIX I

SUPPLEMENTARY MATERIAL FOR THE PAPER TITLED PURGE-GATE: BACKPROPAGATION-FREE TEST-TIME ADAPTATION FOR POINT CLOUD CLASSIFICATION VIA TOKEN PURGING

1. Effect of Batch Size

Figure I-1 shows that increasing the test-time batch size B improves accuracy for both methods, but PG-SP consistently outperforms TENT across all B . Unlike TENT, which struggles with small batches, PG-SP maintains high accuracy even in low-resource settings, demonstrating robustness and adaptability. This highlights a key advantage of PG-SP: it performs well without relying on large batch sizes, making it more practical for real-world deployment under memory or computational constraints. This evaluation is performed on the ShapeNet-C dataset.

2. Real-World ScanObjectNN Variants

Table I-1 confirms that our **PG-SP** strategy improves performance even under genuine domain shifts that are free of synthetic corruptions. Starting from the OBJ-ONLY model used in Table 1 of the main paper, we deploy PG-SP on two challenging variants: *OBJ-BG*, which introduces cluttered backgrounds, and *PB-T50-RS*, which combines point-based sampling and rotation. PG-SP raises accuracy from **74.18%** to **75.56%** on OBJ-BG (**+1.38%**), and from **56.77%** to **61.07%** on PB-T50-RS (**+4.30%**). These gains demonstrate that our token-purging mechanism benefits real-world scenarios beyond the hand-crafted noise studied in ScanObjectNN-C.

3. Dynamics of Purging, Entropy, and Final Accuracy

Effect of purging length L_{pg} on accuracy under *Distortion* corruption. Figure I-4 presents Top-1 accuracy (black, left axis) and logit entropy (red, right axis) for 128 independent runs spanning purging lengths $L_{pg} \in [0, 127]$. Accuracy starts high at $L_{pg} = 0$ (59%), climbs slightly to a plateau of 63% around $L_{pg} \approx 48$, then drops sharply once purging exceeds 80, while entropy steadily rises and spikes when informative tokens are removed. The green dashed line indicates

the maximum accuracy achieved by an exhaustive search restricted to $L_{\text{pg}} \in [0, 32]$. The yellow band marks the six candidate purging lengths

$$\mathcal{L}_{\text{pg}} = \{0, 2, 4, 8, 16, 32\} \quad (\text{A I-1})$$

examined by our entropy-guided schedule; from this sparse set, it selects L_{pg}^* and attains 62.48% accuracy (orange dashed line), essentially matching the global optimum without exhaustive tuning, thereby validating entropy as a practical proxy for near-optimal hyperparameter selection.

Dynamics of purging: interplay between accuracy and entropy (no L_{pg} selection). Figure I-5 traces Top-1 accuracy (black, left axis) and logit entropy (red, right axis) for *Background* corruption, severity level 3, as the purging length L_{pg} sweeps the full range $[0, 127]$; unlike earlier plots, the result of our entropy-guided schedule is *not* shown. Accuracy and entropy evolve inversely: starting at $L_{\text{pg}} = 0$, accuracy is low (27%) while entropy is high (0.60). Incrementally removing the most corrupted tokens ($0 < L_{\text{pg}} \lesssim 64$) steadily boosts accuracy to a plateau of 64% and drives entropy down to 0.25, indicating increased model confidence. Beyond $L_{\text{pg}} \gtrsim 90$, further purging excises informative tokens; entropy rebounds and accuracy collapses, exposing the regime where pruning becomes detrimental. The green dashed line marks the best score achievable by an exhaustive yet *narrow* search over $L_{\text{pg}} \in [0, 32]$, yielding only 38%—well below the true optimum around $L_{\text{pg}} \approx 64$. These dynamics underline the need for a principled, entropy-aware selection strategy: without it, practitioners risk settling on sub-optimal hyperparameters or over-purging, both of which degrade performance.

Purging dynamics under *Background* corruption, severity 7 (no L_{pg} selection shown). Figure I-6 plots Top-1 accuracy (black, left axis) and logit entropy (red, right axis) across 128 purging lengths $L_{\text{pg}} \in [0, 127]$ under *Background* corruption, severity level 7. At $L_{\text{pg}} = 0$, heavy corruption yields low accuracy (13%) and high entropy (0.83). Removing a handful of tokens ($0 < L_{\text{pg}} \leq 32$, yellow band) barely helps—the best accuracy reachable in this restricted range (green dashed line) is only 16%. As purging continues beyond $L_{\text{pg}} \approx 40$, accuracy climbs almost linearly while entropy declines, peaking near $L_{\text{pg}} \approx 112$ with 33% accuracy and 0.45 entropy.

Table-A I-1 Top-1 classification accuracy (%) on real-world ScanObjectNN variants without synthetic corruptions

Variant	Source Only	PG-SP (Ours)	Δ
OBJ-BG	74.18	75.56	+1.38
PB-T50-RS	56.77	61.07	+4.30

Past this point the trends reverse: over-purging ($L_{pg} > 120$) discards informative tokens, causing accuracy to collapse and entropy to rebound. These dynamics reveal that severe corruption demands far more aggressive purging than the conventional search window $L_{pg} \in [0, 32]$ can capture; without an adaptive, entropy-aware mechanism, practitioners risk selecting sub-optimal hyperparameters that leave most potential performance untapped.

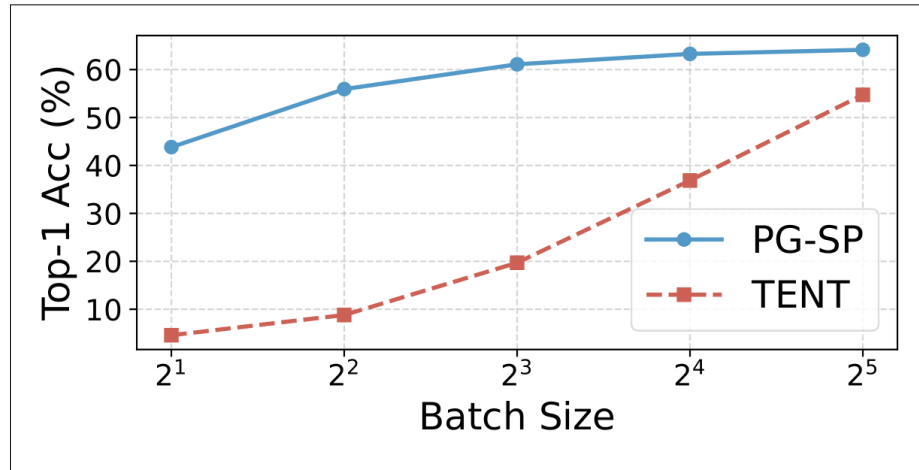


Figure-A I-1 Effect of batch size B on Top-1 accuracy of PG-SP and TENT on the ShapeNet-C dataset

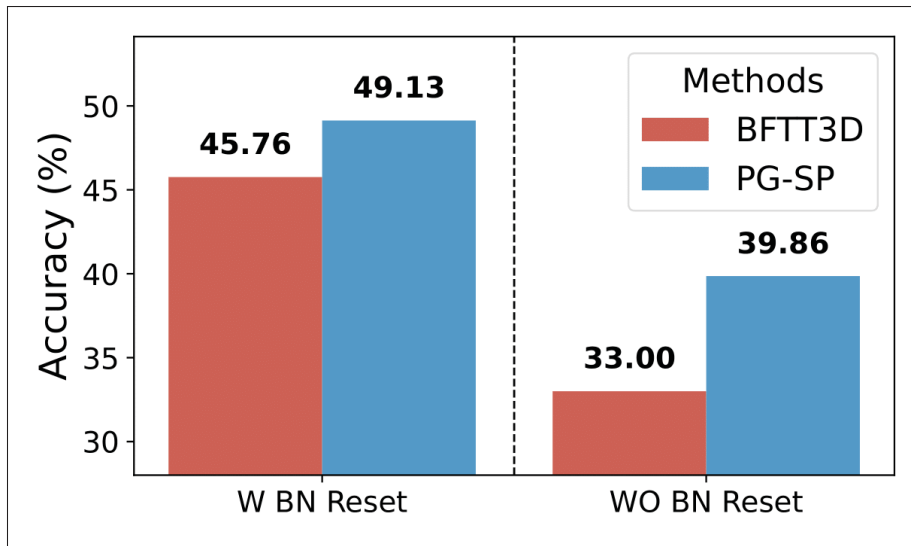


Figure-A I-2 Effect of BatchNorm reset on PG-SP and the BFTT3D baseline on the ScanObjectNN-C dataset. Values are mean Top-1 accuracy

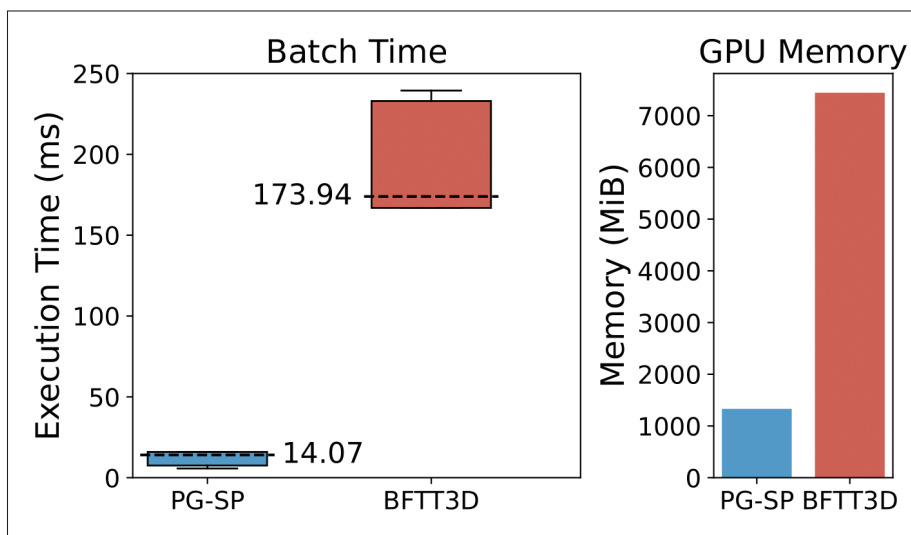


Figure-A I-3 Comparison of GPU time and memory for PG-SP and the BFTT3D baseline. Both are evaluated with batch size $B = 32$ on the ModelNet-C dataset

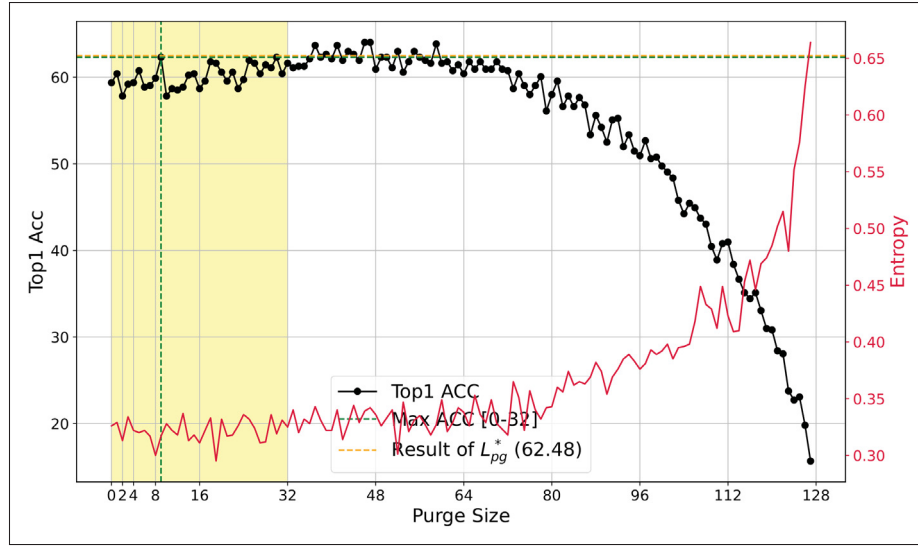


Figure-A I-4 Effect of purging length L_{pg} on Top-1 accuracy for **Distortion** corruption on the ScanObjectNN-C dataset

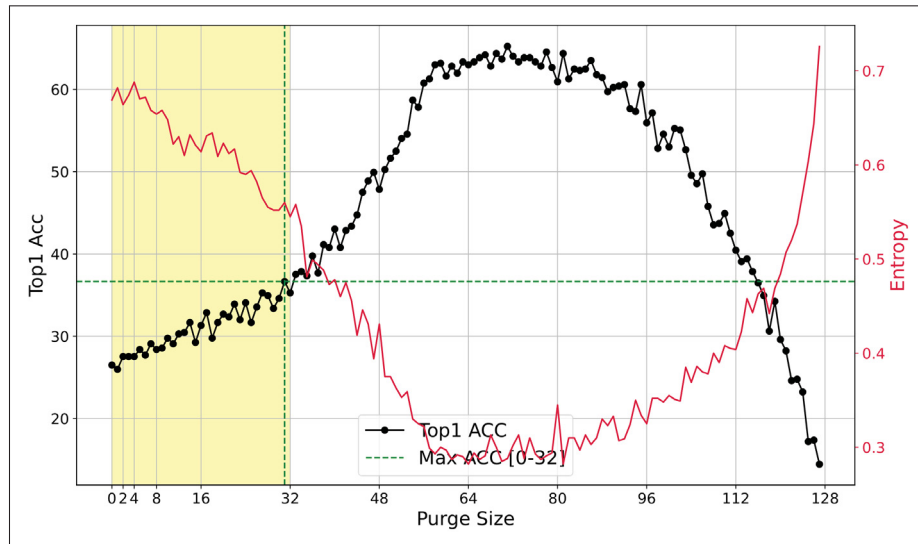


Figure-A I-5 Effect of purging length L_{pg} on Top-1 accuracy for **Background** corruption, level 3, on the ScanObjectNN-C dataset

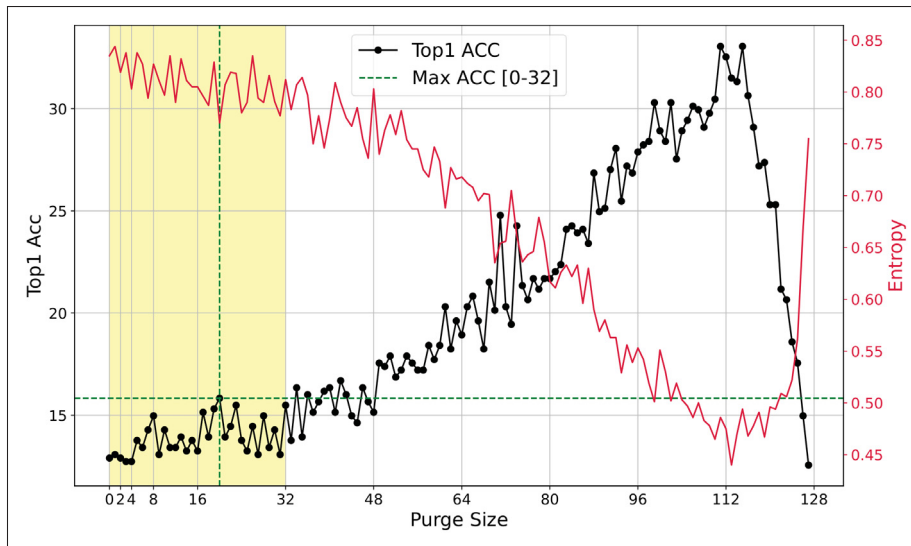


Figure-A I-6 Effect of purging length L_{pg} on Top-1 accuracy for **Background** corruption, **level 7**, on the ScanObjectNN-C dataset

APPENDIX II

SUPPLEMENTARY MATERIAL FOR THE PAPER TITLED REVISITING LAYER NORMALIZATION FOR POINT CLOUD TEST TIME ADAPTATION

1. Method: Proofs

Proof of Theorem 3.3.1. Assume either $\sigma_d(x_t) > 0$ or $\varepsilon > 0$.

Shift. Since $\mathbf{C}(x_t + b \mathbf{1}) = \mathbf{C}x_t$ and $\sigma_d^2(x_t + b \mathbf{1}) = \sigma_d^2(x_t)$ (by Equation (3.10)),

$$\widehat{\mathbf{LN}}(x_t + b \mathbf{1}) = \widehat{\mathbf{LN}}(x_t). \quad (\text{A II-1})$$

Scale. Let $r := d^{-1} \|\mathbf{C}x_t\|^2$ and $f(a; x_t) := \frac{a\sqrt{r + \varepsilon}}{\sqrt{a^2 r + \varepsilon}}$. From Equation (3.11),

$$\widehat{\mathbf{LN}}(a x_t) = \sqrt{d} \frac{a \mathbf{C}x_t}{\sqrt{a^2 \|\mathbf{C}x_t\|^2 + d \varepsilon}} = \frac{a}{|a|} \widehat{\mathbf{LN}}(x_t) f(|a|; x_t). \quad (\text{A II-2})$$

Hence for $a > 0$ we obtain Equation (3.14), and for $a < 0$ we obtain Equation (3.15). For the post-affine output $y(x_t) = \Gamma \widehat{\mathbf{LN}}(x_t) + \beta$,

$$y(a x_t) = \Gamma \widehat{\mathbf{LN}}(a x_t) + \beta = \frac{a}{|a|} f(|a|; x_t) (y(x_t) - \beta) + \beta. \quad (\text{A II-3})$$

Proof of Theorem 3.3.2. Let $h = \widehat{\mathbf{LN}}(x_t)$; componentwise,

$$h_j = \frac{(x_t)_j - \mu_d(x_t)}{\sqrt{\sigma_d^2(x_t) + \varepsilon}}. \quad (\text{A II-4})$$

Since $\sum_{j=1}^d ((x_t)_j - \mu_d(x_t)) = 0$, we have $\frac{1}{d} \sum_{j=1}^d h_j = 0$. For the second moment,

$$\frac{1}{d} \sum_{j=1}^d h_j^2 = \frac{1}{d} \cdot \frac{\sum_{j=1}^d ((x_t)_j - \mu_d(x_t))^2}{\sigma_d^2(x_t) + \varepsilon} = \frac{\sigma_d^2(x_t)}{\sigma_d^2(x_t) + \varepsilon}, \quad (\text{A II-5})$$

which equals 1 when $\varepsilon = 0$, matching the norm identity in Equation (3.13).

Proof of Theorem 3.3.3. Step 1 (token-wise consistency of the random normalization). For batch element b (with fixed but arbitrary token index t) define

$$\mu_{b,t}^{(d)} = \frac{1}{d} \sum_{i=1}^d (x_{b,t}^{(d)})_i, \quad v_{b,t}^{(d)} = \frac{1}{d} \sum_{i=1}^d ((x_{b,t}^{(d)})_i - \mu_{b,t}^{(d)})^2, \quad (\text{A II-6})$$

and the j -th pre-affine normalized coordinate

$$(H_{b,t}^{(d)})_j = \frac{(x_{b,t}^{(d)})_j - \mu_{b,t}^{(d)}}{\sqrt{v_{b,t}^{(d)}}}. \quad (\text{A II-7})$$

Finite fourth moments and the assumed LLN across coordinates imply $\mu_{b,t}^{(d)} \xrightarrow{a.s.} \mu$ and $v_{b,t}^{(d)} \xrightarrow{a.s.} \sigma^2$ as $d \rightarrow \infty$. Let $g(y, a, b) = (y - a)/\sqrt{b}$, continuous on $\{b > 0\}$. By the continuous mapping theorem,

$$(H_{b,t}^{(d)})_j = g((x_{b,t}^{(d)})_j, \mu_{b,t}^{(d)}, v_{b,t}^{(d)}) \Rightarrow \frac{(x_{b,t}^{(d)})_j - \mu}{\sigma}. \quad (\text{A II-8})$$

Uniform integrability (from finite fourth moments) yields moment convergence: $\mathbb{E}((H_{b,t}^{(d)})_j) \rightarrow 0$, $\text{Var}((H_{b,t}^{(d)})_j) \rightarrow 1$.

Step 2 (batch LLN at fixed d). For each d , the variables $\{(H_{b,t}^{(d)})_j\}_{b=1}^{B(d)}$ are i.i.d. with finite variance. Hence,

$$\begin{aligned} \overline{H}_{j,B(d)}^{(d)} &\xrightarrow{a.s.} \mathbb{E}(H_{1,t}^{(d)})_j, \\ (S_{j,B(d)}^2)^{(d)} &\xrightarrow{a.s.} \text{Var}((H_{1,t}^{(d)})_j), \end{aligned} \quad B(d) \rightarrow \infty. \quad (\text{A II-9})$$

Step 3 (diagonal limit). From Step 1, $\mathbb{E}(H_{1,t}^{(d)})_j \rightarrow 0$ and $\text{Var}((H_{1,t}^{(d)})_j) \rightarrow 1$ as $d \rightarrow \infty$. Combining with Step 2 and applying a diagonal argument (or Chebyshev in probability) yields

$$\overline{H}_{j,B(d)}^{(d)} \xrightarrow{\mathbb{P}} 0, \quad (S_{j,B(d)}^2)^{(d)} \xrightarrow{\mathbb{P}} 1, \quad (\text{A II-10})$$

along any sequence $B(d) \rightarrow \infty$.

Corollary 1.1 (Post-affine batch limits and $\varepsilon \geq 0$). *Let $H_{b,t} = \widehat{\text{LN}}(x_{b,t})$ and $y_{b,t} = \Gamma H_{b,t} + \beta$ with $\Gamma = \text{diag}(\gamma)$. Under the assumptions of Theorem 3.3.3, as $d \rightarrow \infty$ and $B = B(d) \rightarrow \infty$,*

$$\overline{y}_{j,B(d)} \xrightarrow{\mathbb{P}} \beta_j, \quad S_{y,j,B(d)}^2 \xrightarrow{\mathbb{P}} \gamma_j^2 \frac{\sigma^2}{\sigma^2 + \varepsilon}, \quad (\text{A II-11})$$

where $\overline{y}_{j,B} = \frac{1}{B} \sum_{b=1}^B (y_{b,t})_j$ and $S_{y,j,B}^2 = \frac{1}{B} \sum_{b=1}^B ((y_{b,t})_j - \overline{y}_{j,B})^2$.

Proof. By the LLN across coordinates, $\mu_{b,t}^{(d)} \rightarrow \mu$ and $v_{b,t}^{(d)} \rightarrow \sigma^2$ a.s. as $d \rightarrow \infty$. Hence $(H_{b,t})_j = ((x_{b,t})_j - \mu_{b,t}^{(d)}) / \sqrt{v_{b,t}^{(d)} + \varepsilon} \Rightarrow (X - \mu) / \sqrt{\sigma^2 + \varepsilon}$, so $\mathbb{E}(H_{b,t})_j \rightarrow 0$ and $\text{Var}(H_{b,t})_j \rightarrow \sigma^2 / (\sigma^2 + \varepsilon)$. Applying the batch LLN and mapping through $y_j = \gamma_j H_j + \beta_j$ gives the limits.

2. Experiments

To test how our theoretical analysis of LN manifests in practice, we first evaluate the effect of the LN layer on domain information, followed by our backpropagation-free LN adaptation on three corrupted 3D point-cloud suites, ModelNet40-C Sun *et al.* (2022), ShapeNet-C Chang *et al.* (2015), and ScanObjectNN-C Uy *et al.* (2019), following test-time protocols used in prior work Wang *et al.* (2024c); Mirza *et al.* (2023). These benchmarks introduce diverse perturbations (e.g., noise, density shifts, geometric distortions) to stress robustness and cross-domain generalization. We report results under both *episodic* (reset after each test batch) and *online* (continuous stream) adaptation; because our method is a purely forward, stateless reparameterization of the LN affine (no gradients, no optimizers, no BN buffers), predictions are *identical* under the two protocols, there is no weight/buffer mutation and thus no temporal overfitting. For fair comparison, we start from the publicly released PointMAE backbone Pang *et al.* (2022) as prepared in MATE Mirza

et al. (2023) and reproduce baselines within the same evaluation harness Wang *et al.* (2024c); Mirza *et al.* (2023). We follow TENT in resetting the BN statistics. Adaptation employs a batch size of $B = 32$, with a single forward pass per batch (no data augmentation or auxiliary adaptation passes), introduces no additional hyperparameters, and incurs only negligible compute/memory overhead; experiments are run on a single RTX 2060 GPU.

2.1 Datasets

We evaluate on three corrupted point-cloud benchmarks that share the same set of 15 test-time corruption types, covering geometric transformations, additive noise, and density variations, implemented by Sun *et al.* Sun *et al.* (2022), which enables apples-to-apples robustness comparisons across datasets. **ModelNet40-C** Sun *et al.* (2022) augments the original ModelNet40 test split with these 15 corruptions to probe resilience to realistic acquisition artifacts (e.g., sensor noise and LiDAR sparsity). **ShapeNet-C** is derived from ShapeNetCore-v2 Chang *et al.* (2015) (51,127 shapes, 55 categories; 70%/10%/20% train/val/test) by applying the same 15 corruptions to the test set following the open-source pipeline of Sun *et al.* (2022), as introduced in Mirza *et al.* (2023). Finally, **ScanObjectNN-C** builds on the real-world ScanObjectNN dataset Uy *et al.* (2019) (15 classes; 2,309 train / 581 test) by corrupting only the test set with the same 15 object-level perturbations, following the procedure of Mirza *et al.* (2023).

2.2 Batch-size Ablations

We vary the test-time batch size B to assess practical sensitivity under a low-batch regime. **LN-TTA** (ours) shows a smooth, monotonic gain with B and, crucially, retains strong accuracy at small B , where TENT degrades sharply. This behavior stems from our design: since LN enforces token-wise zero mean and unit variance, the *post-affine* statistics are already well concentrated; our single-pass, backprop-free calibration merely recenters and rescales that operating point, requiring far less batch evidence than optimization-based TTA. At $B=32$ (used in Table 2.3), **LN-TTA** attains **64.38%** mean accuracy on ShapeNet-C, outperforming TENT in the standard setting and maintaining a margin even relative to its online variant, all while remaining stateless

(identical “Standard”/“Online” predictions), source-free, and hyperparameter-free. This justifies our choice of $B=32$ as a balanced point between stability, throughput, and memory.

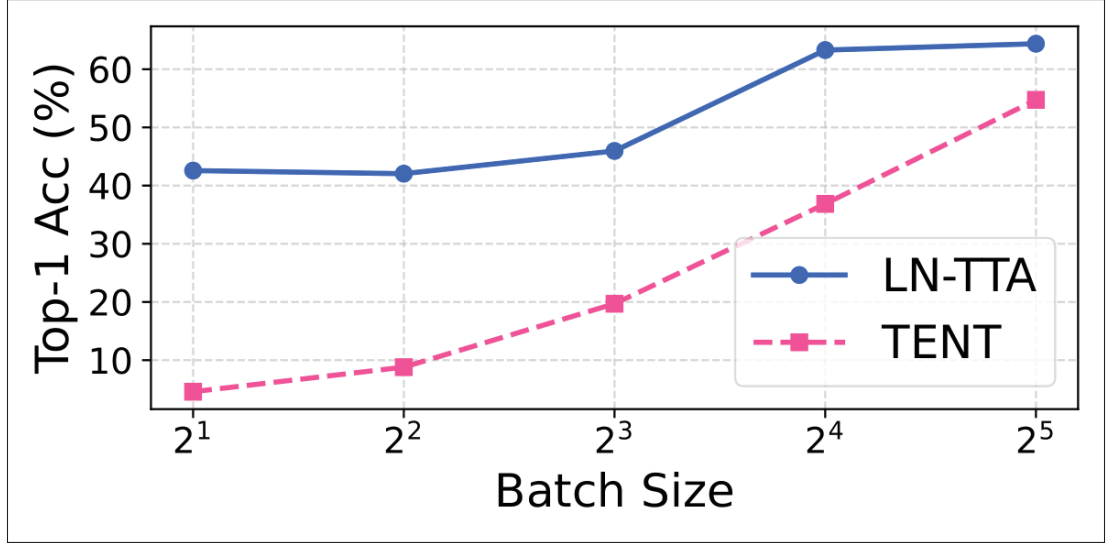


Figure-A II-1 **Batch-size sensitivity on ShapeNet-C.** Top-1 accuracy (%) versus test-time batch size B for **LN-TTA (ours)** and **TENT** under the same backbone and corruption protocol. Both methods improve with larger B , but **LN-TTA** maintains a clear margin across the entire range and is notably more stable in the small- B regime

2.3 Average per-feature marginals before and after LN.

We synthesize batches whose coordinates are drawn from a mix of distributions {Normal, Laplace, Uniform, Student- t , Logistic, Exponential}, compute per-feature histograms across tokens, and average these histograms over features to obtain an “average marginal” (Section 3.3.3 viewpoint). We then compare pre-LN (dashed) and post-LN (solid) curves while (i) sweeping batch size B at fixed d and (ii) sweeping d at fixed B (Figure II-2). Post-LN curves align closely with the standard normal across settings, while pre-LN curves reflect the heterogeneous input laws. This outcome is consistent with our theory: LN enforces token-wise zero mean and unit variance (Theorem 3.3.2), and the per-feature, across-tokens marginals concentrate near $(0, 1)$ as (d, B) grow (Theorem 3.3.3). Consequently, re-standardizing “across the batch” is not the right lever for LN; the meaningful degrees of freedom lie in the angular structure within $\mathcal{H}$ and in the post-affine mean/scale, which motivates our post-affine test-time updates in Section 3.3.4.

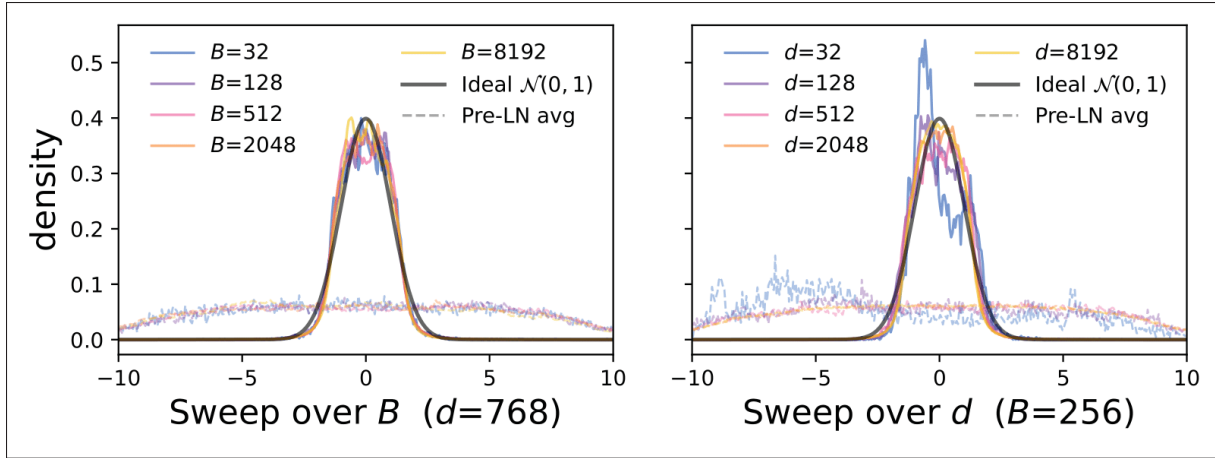


Figure-A II-2 **Pre- vs. post-LN per-feature marginals.** We plot the *average* per-feature marginal density across tokens (dashed = pre-LN, solid = post-LN). **Left:** sweep batch size B at fixed $d=768$. **Right:** sweep feature dimension d at fixed $B=256$. The dashed black curve is the ideal $\mathcal{N}(0, 1)$

2.3.1 Per-channel marginals across batch sizes and feature dimensions.

For each configuration with batch size $B \in \{32, 128, 512, 2048, 8192\}$ and feature dimension $d \in \{32, 128, 512, 2048, 8192\}$, we synthesize inputs whose coordinates are drawn from a heterogeneous mix of distributions {Normal, Laplace, Uniform, Student- t , Logistic, Exponential}; see Figure II-3 to Figure II-7. For every feature, we compute its empirical marginal across tokens and overlay all per-feature curves together with their featurewise average, both *before* and *after* LN. Across all (B, d) pairs, the pre-LN panels exhibit wide, dataset-dependent variability (shifts, scales, and tail behaviours), whereas the post-LN panels collapse tightly toward the standard normal. Two trends are consistent throughout: (i) at fixed d , increasing B concentrates the per-feature, across-tokens marginals around mean 0 and variance 1; and (ii) at fixed B , increasing d further tightens the post-LN dispersion around $\mathcal{N}(0, 1)$. These observations match our analysis: LN enforces token-wise zero mean and unit variance exactly (Theorem 3.3.2), and under mild moment assumptions the per-feature batch marginals concentrate near $(0, 1)$ as $(d, B) \rightarrow \infty$ (Theorem 3.3.3). Consequently, batch/channel re-standardization is not the right lever for LN; the actionable degrees of freedom lie in the mean-free angular structure

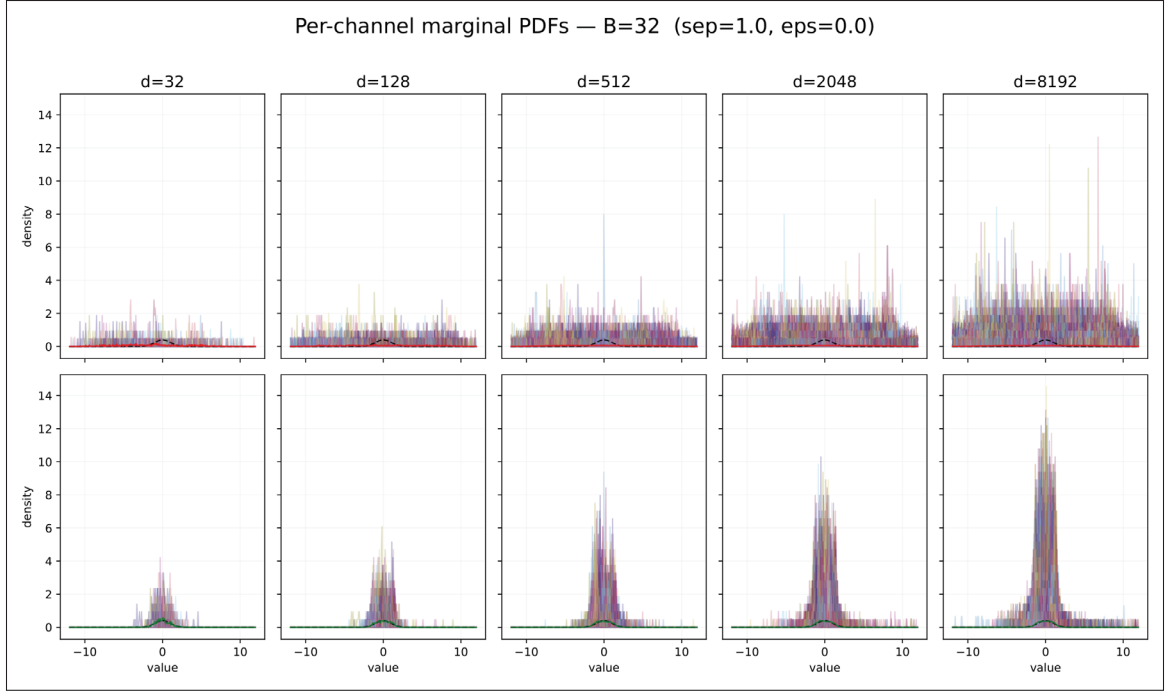


Figure-A II-3 Per-feature marginal PDFs before/after LN across batch sizes and feature dimensions. Each panel (row) fixes a batch size $B \in \{32, 128\}$ and sweeps the feature dimension $d \in \{32, 128, 512, 2048, 8192\}$ across columns. Thin colored curves are per-feature empirical marginals across tokens; solid **red** is the featurewise average *pre*-LN, solid **green** is the average *post*-LN, and the dashed black curve is the ideal $\mathcal{N}(0, 1)$ (here $\varepsilon=0$). Across all (B, d) , pre-LN marginals are heterogeneous, while post-LN marginals collapse toward mean 0 and variance 1, with dispersion shrinking as B and d increase. This visualizes coordinate-wise concentration toward $(0, 1)$ as $(d, B) \rightarrow \infty$ (Theorem 3.3.3)

within $\mathcal{H}$ and in the post-affine mean/scale, which motivates our post-affine test-time updates in Section 3.3.4.

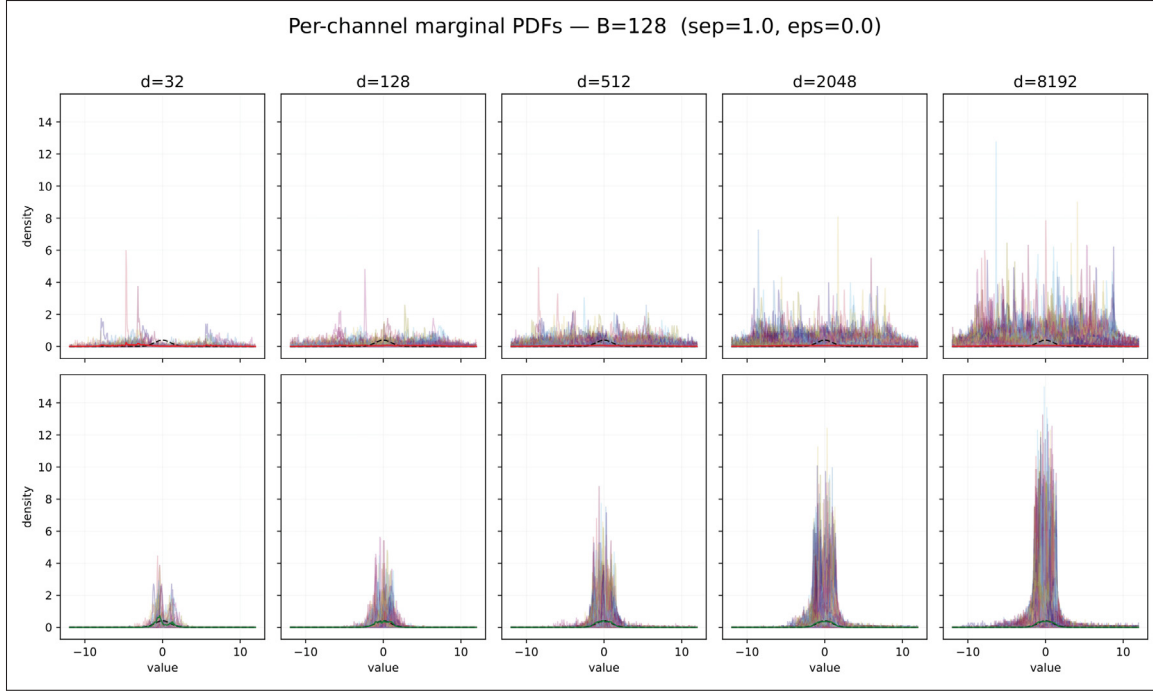


Figure-A II-4 Per-feature marginal PDFs before/after LN across batch sizes and feature dimensions. Each panel (row) fixes a batch size $B \in \{32, 128\}$ and sweeps the feature dimension $d \in \{32, 128, 512, 2048, 8192\}$ across columns. Thin colored curves are per-feature empirical marginals across tokens; solid **red** is the featurewise average *pre*-LN, solid **green** is the average *post*-LN, and the dashed black curve is the ideal $\mathcal{N}(0, 1)$ (here $\varepsilon=0$). Across all (B, d) , pre-LN marginals are heterogeneous, while post-LN marginals collapse toward mean 0 and variance 1, with dispersion shrinking as B and d increase. This visualizes coordinate-wise concentration toward $(0, 1)$ as $(d, B) \rightarrow \infty$ (Theorem 3.3.3)

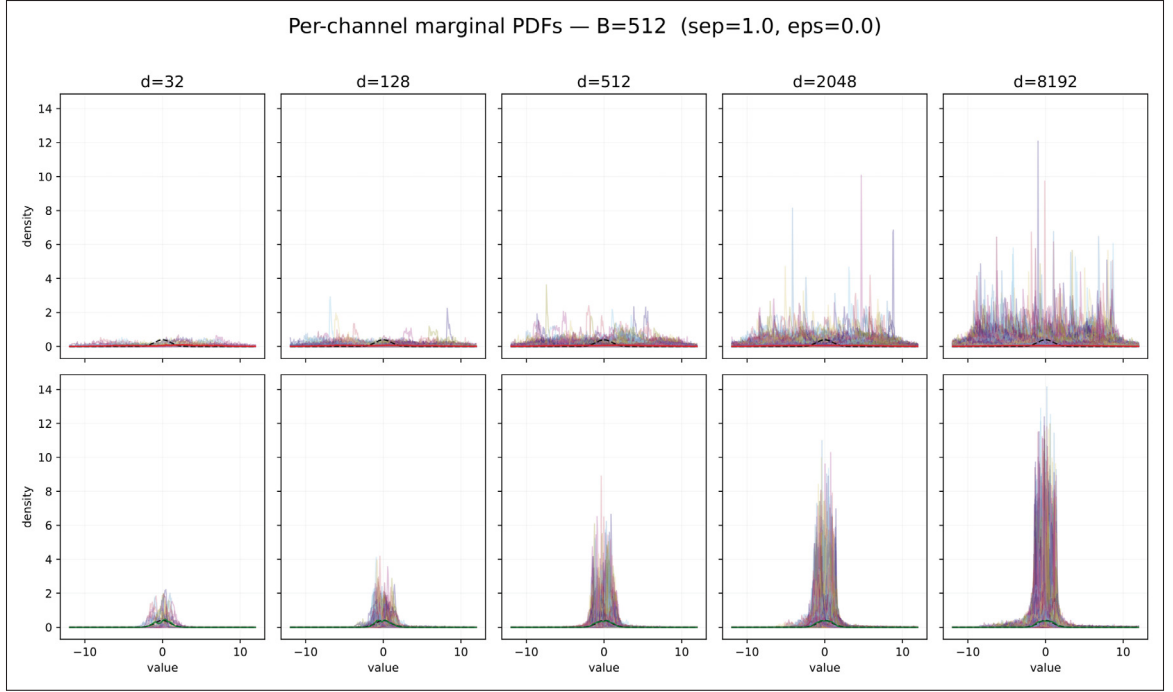


Figure-A II-5 Per-feature marginal PDFs before/after LN across batch sizes and feature dimensions. Each panel (row) fixes a batch size $B \in \{512, 2048\}$ and sweeps the feature dimension $d \in \{32, 128, 512, 2048, 8192\}$ across columns. Thin colored curves are per-feature empirical marginals across tokens; solid **red** is the featurewise average *pre*-LN, solid **green** is the average *post*-LN, and the dashed black curve is the ideal $\mathcal{N}(0, 1)$ (here $\varepsilon=0$). Across all (B, d) , pre-LN marginals are heterogeneous, while post-LN marginals collapse toward mean 0 and variance 1, with dispersion shrinking as B and d increase. This visualizes coordinate-wise concentration toward $(0, 1)$ as $(d, B) \rightarrow \infty$ (Theorem 3.3.3)

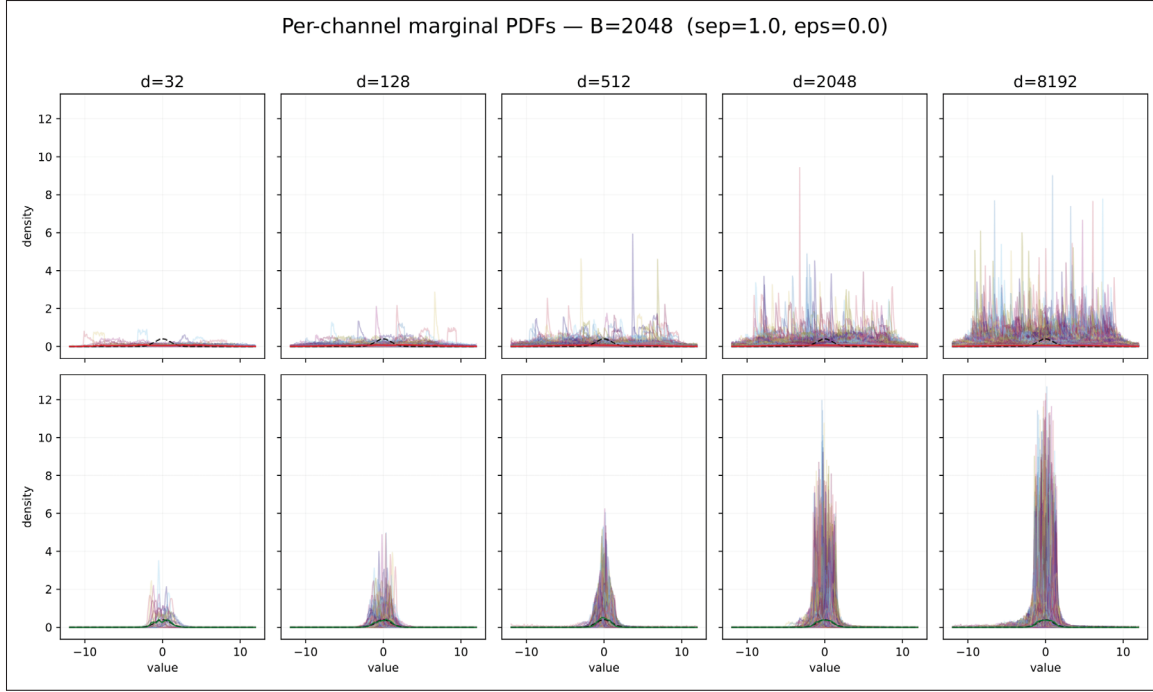


Figure-A II-6 Per-feature marginal PDFs before/after LN across batch sizes and feature dimensions. Each panel (row) fixes a batch size $B \in \{512, 2048\}$ and sweeps the feature dimension $d \in \{32, 128, 512, 2048, 8192\}$ across columns. Thin colored curves are per-feature empirical marginals across tokens; solid red is the featurewise average *pre*-LN, solid green is the average *post*-LN, and the dashed black curve is the ideal $\mathcal{N}(0, 1)$ (here $\varepsilon=0$). Across all (B, d) , pre-LN marginals are heterogeneous, while post-LN marginals collapse toward mean 0 and variance 1, with dispersion shrinking as B and d increase. This visualizes coordinate-wise concentration toward $(0, 1)$ as $(d, B) \rightarrow \infty$ (Theorem 3.3.3)

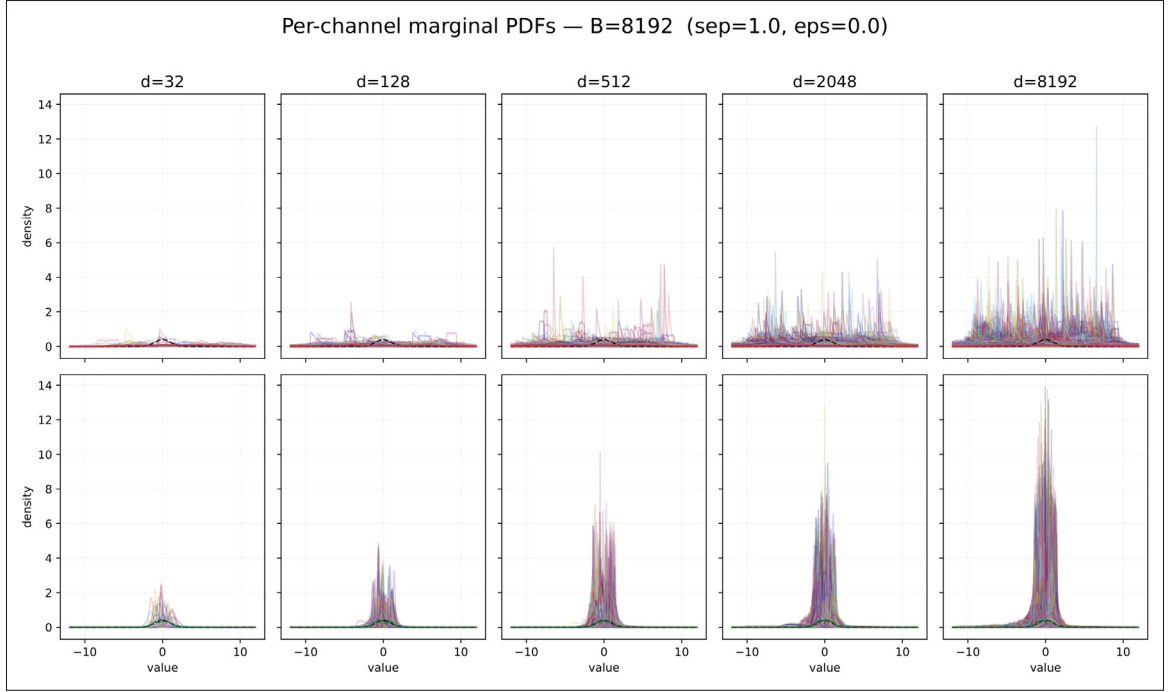


Figure-A II-7 Per-feature marginal PDFs before/after LN across batch sizes and feature dimensions. Each panel (row) fixes a batch size $B = 8192$ and sweeps the feature dimension $d \in \{32, 128, 512, 2048, 8192\}$ across columns. Thin colored curves are per-feature empirical marginals across tokens; solid **red** is the featurewise average *pre*-LN, solid **green** is the average *post*-LN, and the dashed black curve is the ideal $\mathcal{N}(0, 1)$ (here $\varepsilon=0$). Across all (B, d) , *pre*-LN marginals are heterogeneous, while *post*-LN marginals collapse toward mean 0 and variance 1, with dispersion shrinking as B and d increase. This visualizes coordinate-wise concentration toward $(0, 1)$ as $(d, B) \rightarrow \infty$ (Theorem 3.3.3)

APPENDIX III

SUPPLEMENTARY MATERIAL FOR THE PAPER TITLED PRISMA: PROMPT REFINEMENT FOR INSTANCE SEGMENTATION OF MODEL-BASED UNSEEN OBJECTS

The code for this paper is provided in this repository (https://github.com/MosyMosy/self_grounding_sam6d).

1. Method

1.1 Model-Based Prompt Generation (MPG)

In this section, we present the detailed pseudo-code for our MPG module, as outlined in Algorithm III-1. The pseudo-code captures the core computational flow and key operations of the module, including scene descriptor extraction, object template processing, similarity computation, and the generation of foreground prompts. These steps are designed to efficiently identify and focus on relevant regions of the input scene, minimizing unnecessary computation while maintaining high segmentation quality.

1.2 Self-Grounding (SG)

Similar to the MPG module, the prompt generation process in our SG module relies on the similarity between the scene’s patches and the object’s embeddings. However, the key distinction lies in the design of the attention layer, which introduces a more effective approach to grouping and processing tokens based on their similarity scores. This mechanism is detailed by the pseudo-code in Algorithm III-2, and matches the Self-Grounding formulation of Section 4.3.4.

Theoretical Support for Self-Grounding Attention To complement the intuition presented in Section 4.3.4, we now provide a formal justification of why Self-Grounding improves both accuracy and efficiency. Specifically, we show that the routing mask $\mathbf{m}^i$ induces a block-diagonal attention structure that provably suppresses cross-group interference while reducing

Algorithm-A III-1 Foreground Prompt Generation (MPG)

Require: Input image I , threshold τ_+

Ensure: Foreground prompt locations $\mathcal{P}_S$, binary mask $\mathbf{m}_S$

- 1: $\mathbf{F}_S^\ell \leftarrow \text{encode_full_size}(I)$
- 2: $\tilde{\mathbf{F}}_O^\ell \leftarrow \text{ref_data}[\text{last_token}]$
- 3: $\mathbf{m}_T \leftarrow (\tilde{\mathbf{F}}_O^\ell.\text{sum}(-1) > 0)$
- 4: $\mathbf{f}_+^{(\text{raw})} \leftarrow \tilde{\mathbf{F}}_O^\ell.\text{sum}(-2) / \mathbf{m}_T.\text{sum}(-1)[:, \text{None}]$
- 5: $\mathbf{f}_+^{(\text{raw})}[\mathbf{m}_T.\text{sum}(-1) == 0] \leftarrow 0$
- 6: $\mathbf{f}_+ \leftarrow \text{mean}(\mathbf{f}_+^{(\text{raw})}[:, :, \text{num_templates}], \text{dim} = (0, 1))$
- 7: $\mathbf{F}_S^\ell \leftarrow \mathbf{F}_S^\ell / \|\mathbf{F}_S^\ell\|_2$
- 8: $\mathbf{f}_+ \leftarrow \mathbf{f}_+ / \|\mathbf{f}_+\|_2$
- 9: $\mathbf{S} \leftarrow \mathbf{F}_S^\ell \mathbf{f}_+^\top$
- 10: $\mathbf{S} \leftarrow (\mathbf{S} - \mathbf{S}.\text{min}()) / (\mathbf{S}.\text{max}() - \mathbf{S}.\text{min}())$
- 11: $\mathcal{G} \leftarrow \text{gen_patch_grid_points}(\text{output_size}, \text{patch_size})$
- 12: $\mathbf{m}_S, \mathcal{P}_S \leftarrow \text{sim_2_point_prompts}(\mathbf{S}, \mathcal{G}, \text{output_size}, \tau_+)$
- 13: **return** $\mathcal{P}_S, \mathbf{m}_S$

computational complexity. The following propositions establish these guarantees under mild assumptions, and connect them directly to the practical benefits observed in our experiments.

Let

$$A^i = \text{softmax}\left(\frac{\mathbf{Q}_S^i (\mathbf{K}_S^i)^\top}{\sqrt{d}}\right) \quad (\text{A III-1})$$

denote the pre-SG attention at layer i , where $\mathbf{Q}_S^i, \mathbf{K}_S^i \in \mathbb{R}^{N \times d}$ are the scene queries and keys. In the unconstrained case, A^i is fully dense.

After applying the routing mask $\mathbf{m}^i \in \{0, 1\}^N$ (constructed by thresholding similarity to the object prototype), the effective attention becomes block diagonal:

$$\tilde{A}^i = \text{blkdiag}(A_{++}^i, A_{--}^i), \quad (\text{A III-2})$$

where A_{++}^i denotes attention restricted to indices with $\mathbf{m}^i = 1$ (foreground), and A_{--}^i to those with $\mathbf{m}^i = 0$ (background). The cross-group terms A_{+-}^i and A_{-+}^i vanish; their total contribution

can be measured by the Frobenius energy

$$\mathcal{E}_\times^i = \|A_{+-}^i\|_F^2 + \|A_{-+}^i\|_F^2. \quad (\text{A III-3})$$

Proposition 1 (Cross-group interference reduction). If $\mathbf{m}^i$ is constructed by thresholding a similarity score that is stochastically larger for foreground than for background queries (monotone likelihood ratio property), then

$$\mathbb{E}[\mathcal{E}_\times^i] \quad (\text{A III-4})$$

decreases monotonically in τ_G across its feasible range, with equality only in the degenerate cases where all tokens are classified as foreground or all as background.

Proof sketch. Thresholding a monotone likelihood ratio statistic corresponds to the Neyman–Pearson most powerful test. As τ_G increases, fewer ambiguous tokens fall into the “mixed” zone, reducing the expected mass of the off-diagonal blocks. Hence $\mathbb{E}[\mathcal{E}_\times^i]$ decreases monotonically, strictly until one group is exhausted.

Proposition 2 (Attention compute reduction). Let N_+ and N_- be the sizes of the two groups with $N_+ + N_- = N$. Dense attention costs $\Theta(N^2)$. After grouping, the cost becomes

$$\Theta(N_+^2 + N_-^2), \quad (\text{A III-5})$$

which saves

$$\Theta(2N_+N_-) \quad (\text{A III-6})$$

pairwise interactions and eliminates all cross-group multiplications. When N_+/N is bounded away from 0 and 1, the constant-factor reduction is substantial. Combined with prompt sparsification from MPG, this yields significant end-to-end speedups.

1.3 Confidence-Weighted Scoring (CW)

We use the SAM6D scoring objectives, rewritten in our standardized notation.

Algorithm-A III-2 Self-Grounding Attention Layer

Require: Input $X \in \mathbb{R}^{B \times N \times C}$, optional grounding info $\mathbf{g_info}$, attention bias $\mathbf{attn_bias}$
Ensure: Output $X' \in \mathbb{R}^{B \times N \times C}$, updated grounding info $\mathbf{g_info}'$

- 1: $\mathbf{qkv} \leftarrow \mathbf{qkv}(X)$; reshape to $(B, N, 3, \text{heads}, C/\text{heads})$
- 2: Split into $\mathbf{Q}, \mathbf{K}, \mathbf{V}$
- 3: **if** $\mathbf{g_info} \neq \text{None}$ **then**
- 4: $\mathbf{g_layer} \leftarrow \mathbf{g_info}[:, 0]$; $\mathbf{g_info}' \leftarrow \mathbf{g_info}[:, 1:]$
- 5: $\mathbf{Q}_{G_+} \leftarrow \mathbf{g_layer}[:, 0]$
- 6: /* Reserved tokens and pure tokens */
- 7: $\mathbf{Q}_{\text{pure}}, \mathbf{K}_{\text{pure}}, \mathbf{V}_{\text{pure}} \leftarrow \mathbf{Q}[:, r:], \mathbf{K}[:, r:], \mathbf{V}[:, r:]$
- 8: /* Normalize */
- 9: $\hat{\mathbf{Q}}_{\text{pure}} \leftarrow \mathbf{Q}_{\text{pure}} / \|\mathbf{Q}_{\text{pure}}\|_2$; $\hat{\mathbf{Q}}_G \leftarrow \mathbf{Q}_{G_+} / \|\mathbf{Q}_{G_+}\|_2$
- 10: /* Similarity and mask */
- 11: $\mathbf{s} \leftarrow (\hat{\mathbf{Q}}_G \cdot \hat{\mathbf{Q}}_{\text{pure}}) \cdot \text{sum}(-1)$
- 12: $\mathbf{m}_{\text{fg}}^i \leftarrow (\mathbf{s} > \tau_G)$; $\mathbf{m}_{\text{bg}}^i \leftarrow \neg \mathbf{m}_{\text{fg}}^i$
- 13: /* Group tokens */
- 14: $\{\mathbf{Q}_{\pm}, \mathbf{K}_{\pm}, \mathbf{V}_{\pm}\} \leftarrow \text{masked}(\mathbf{Q}_{\text{pure}}, \mathbf{K}_{\text{pure}}, \mathbf{V}_{\text{pure}})$
- 15: /* Add reserved tokens (e.g., special tokens) */
- 16: $\mathbf{Q}_{\pm} \leftarrow \text{cat}([\mathbf{Q}[:, :r], \mathbf{Q}_{\pm}])$ (similarly for $\mathbf{K}_{\pm}, \mathbf{V}_{\pm}$)
- 17: /* Compute group-wise attention */
- 18: $\mathbf{X}_+ \leftarrow \text{attn}(\mathbf{Q}_+, \mathbf{K}_+, \mathbf{V}_+, \mathbf{attn_bias})$
- 19: $\mathbf{X}_- \leftarrow \text{attn}(\mathbf{Q}_-, \mathbf{K}_-, \mathbf{V}_-, \mathbf{attn_bias})$
- 20: /* Merge by mask */
- 21: $\mathbf{X}[\mathbf{m}_{\text{fg}}^i] \leftarrow \mathbf{X}_+$; $\mathbf{X}[\mathbf{m}_{\text{bg}}^i] \leftarrow \mathbf{X}_-$
- 22: **else**
- 23: $\mathbf{g_info}' \leftarrow \text{None}$; $\mathbf{X} \leftarrow \text{attn}(\mathbf{Q}, \mathbf{K}, \mathbf{V}, \mathbf{attn_bias})$
- 24: **end if**
- 25: Reshape $\mathbf{X} \rightarrow (B, N, C)$;;
- project: $\mathbf{X}' \leftarrow \text{drop}(\text{proj}(\mathbf{X}))$
- 26: **return** $\mathbf{X}', \mathbf{g_info}'$

Semantic Matching Score

$$SC_{\text{sem}} = \frac{1}{K} \sum_{k=1}^K \max_{o=1, \dots, n_O} \left(\max_{c=1, \dots, n_C} \text{Sim}(\mathbf{f}_O^g(o, c), \mathbf{f}_{S_p}^g) \right), \quad (\text{A III-7})$$

where $\mathbf{f}_O^g(o, c) \in \mathbb{R}^d$ is the global (cls) embedding for object O_o and view C_c , and $\mathbf{f}_{S_p}^g \in \mathbb{R}^d$ is the proposal's global embedding.

Appearance Matching Score

$$SC_{\text{app}} = \frac{1}{H'W'} \sum_{hw=1}^{H'W'} \max_o \left(\max_c \text{Sim}(\mathbf{F}_O^\ell(o, c, hw, :), \mathbf{f}_{S_p}^\ell(hw)) \right), \quad (\text{A III-8})$$

with $\mathbf{F}_O^\ell(o, c, hw, :) \in \mathbb{R}^d$ the local patch embedding and $\mathbf{f}_{S_p}^\ell(hw) \in \mathbb{R}^d$ the proposal's patch embedding.

Geometric Matching Score Let $\mathcal{B}_S$ and $\mathcal{B}_O$ denote the bounding boxes of the proposal and the projected object, respectively. Then

$$SC_{\text{geo}} = \frac{|\mathcal{B}_S \cap \mathcal{B}_O|}{|\mathcal{B}_S \cup \mathcal{B}_O|}, \quad SC_{\text{geo}} = r_{\text{vis}} \cdot \frac{|\mathcal{B}_S \cap \mathcal{B}_O|}{|\mathcal{B}_S \cup \mathcal{B}_O|} \text{ (with visibility weight)}. \quad (\text{A III-9})$$

Final Object Matching Score

$$SC_m = \frac{SC_{\text{sem}} + SC_{\text{app}} + r_{\text{vis}} \cdot SC_{\text{geo}}}{1 + 1 + r_{\text{vis}}}. \quad (\text{A III-10})$$

We weight SC_m by the confidence weights CW from the main text:

$$SC_m^{\text{CW}} = SC_m \odot \text{CW}, \quad SC_m, \text{CW} \in \mathbb{R}^{|\mathcal{M}|}, \quad (\text{A III-11})$$

where $|\mathcal{M}|$ is the number of segmentation proposals and $\odot$ is elementwise multiplication.

2. Analysis

2.1 Effect of τ_+ on the Prompt Area

As evident from Figure III-1, a small value of τ_+ removes only a small portion of the scene, leaving a large area to be prompted (green regions). As we increase τ_+ (from 0.2 to 0.6), the filtered area grows, and the prompts better focus on the foreground objects. Finally, at $\tau_+ = 0.6$

(black), almost all background areas are filtered out, making prompting fully focused on the foreground area.

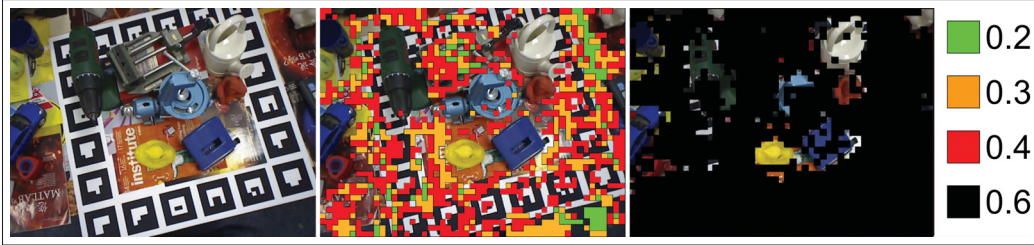


Figure-A III-1 Areas removed by each τ_+ value. The higher the value of τ_+ , the smaller the prompt area. For high similarity thresholds, the prompt area can miss foreground objects, showcasing the necessity of SG to enhance the prompt’s focus area

2.2 Complexity and Implementation Notes

The end-to-end gain arises from two effects. First, MPG reduces the number of SAM queries roughly in proportion to the foreground occupancy α . Second, SG removes cross-group attention multiplications and empirically shrinks compute inside DINOv2 by a constant factor while sharpening features for downstream matching. In our implementation, object features and per-layer grounding tokens ($\mathbf{Q}_{G_+}^i, \mathbf{K}_{G_+}^i, \mathbf{V}_{G_+}^i$) are precomputed offline and cached; scene features are computed once and reused for both MPG and SG. A small set of reserved tokens (e.g., 1–2 special tokens or low-frequency spatial samples) is concatenated to both groups to preserve global context, and a fallback rule disables routing in layers where the foreground mass collapses (see Section 4.3.4). The mechanism is backbone-agnostic (DINOv2-S/B/L, and SAM variants such as FastSAM, SAM-HQ) since no weights are modified.

Resource Analysis of SG Attention We analyze inference time and peak memory usage of different DINOv2 models with and without SG attention, showing minimal overhead (Figure III-2). With total inference around 770 ms per scene, the SG layer adds minor time for larger models and can even reduce time for smaller ones. Peak memory remains close to standard DINOv2 across sizes.

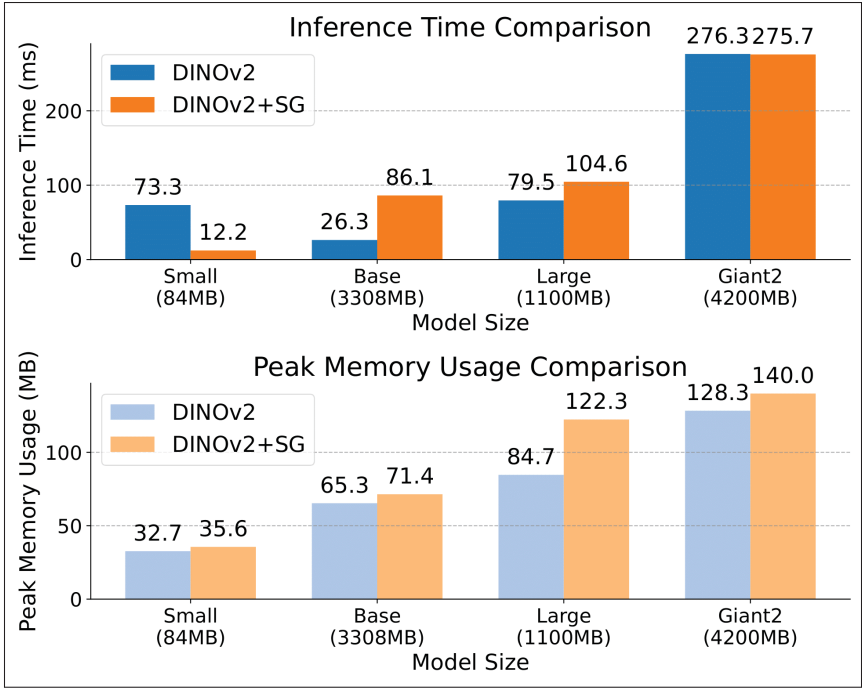


Figure-A III-2 Effect of SG attention on compute and memory of DINOv2. Input 224×224, batch size 10, averaged over 10 runs. Numbers in parentheses are model sizes

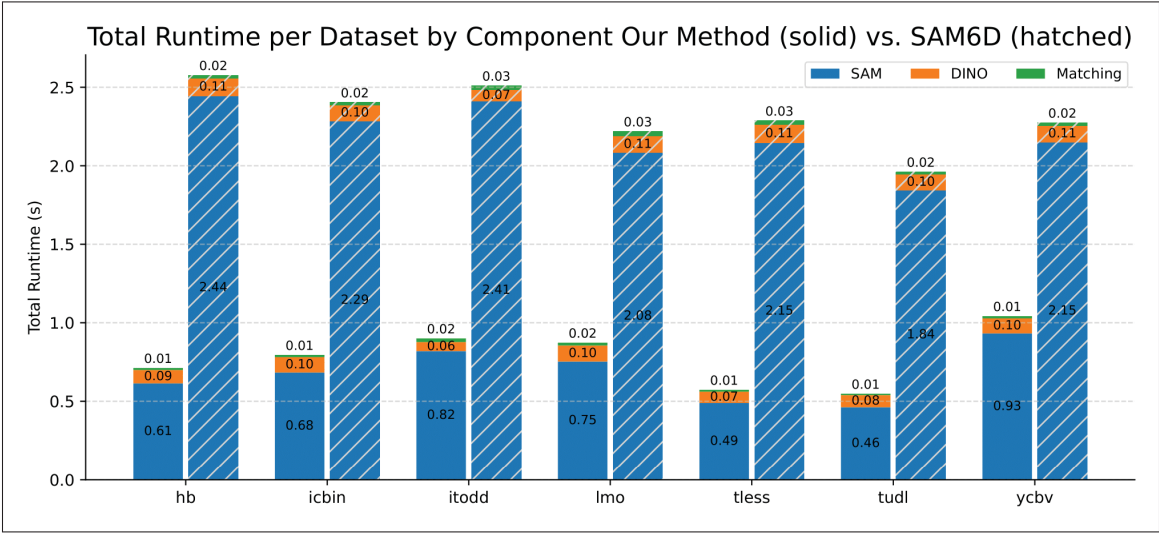


Figure-A III-3 Component-wise runtime: our method vs. grid-based full prompting (SAM6D)

2.3 Effect of Number of Views ($|\mathbb{C}|$) on Average Precision for MISUO

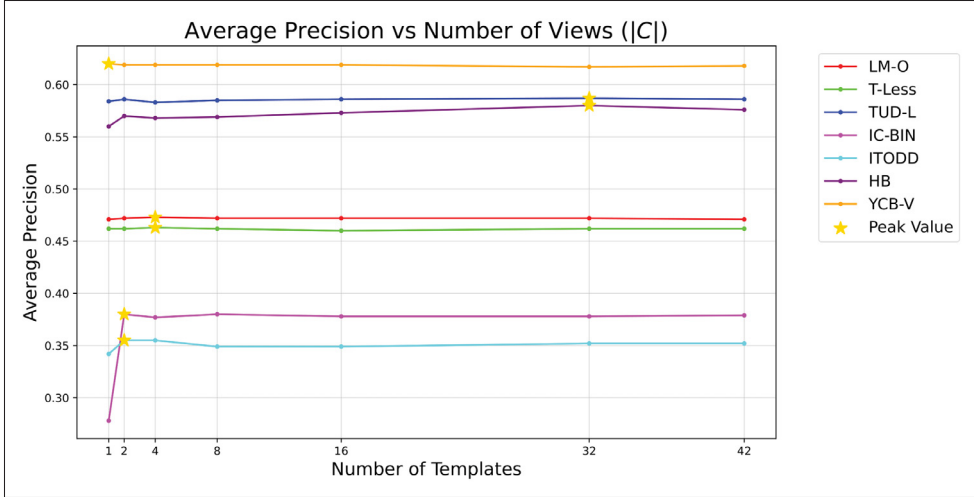


Figure-A III-4 Effect of the number of views on AP across datasets. Peaks are starred, showing optimal counts

In real-world scenarios, acquiring multi-view images of target objects O can be time-consuming. Hence fewer views ease deployment. We analyze the sensitivity of our prompt generation to the number of views in the multi-view representation $\mathbb{M}_{\mathbb{C}}^O$. From Figure III-4, we observe:

LM-O: AP starts at 0.471 (one view) and plateaus near four views (0.473), indicating limited gains beyond a few views.

T-LESS: Minimal variation; peak 0.463 at four views; slight decline later—additional views add redundancy.

TUD-L: Small, steady gains to 0.587 at 32 views—more views capture variability, with diminishing returns.

IC-BIN: Big jump from 0.278 (one) to 0.380 (two), then saturation—early diversity matters most.

ITODD: Improves from 0.342 (one) to 0.355 (two), then flat/slight drop—extra views may add noise.

HB: Rises from 0.560 (one) to 0.580 (32), slight dip at 42—useful but saturating.

YCB-V: Stable near 0.620 regardless of view count—low variability or high robustness.

General Trends:

- Diminishing returns after 2–4 views for most datasets.
- High-variability datasets (e.g., TUD-L, HB) benefit more from additional views.
- Excessive views can slightly hurt (noise/redundancy) on some datasets.

2.4 Prompt Quality PQ Comparison Between SG and Full-Scene Prompting

We define prompt quality as $PQ = \frac{|TP|}{|FP|}$, where TP counts prompts that hit target objects, and FP counts background hits. Comparing our method (SG) to full-scene prompting (SAM6D), the quality ratio is:

$$PQ_r = \frac{TP_{SG} \times FP_{SAM6D}}{TP_{SAM6D} \times FP_{SG}}. \quad (\text{A III-12})$$

Figure III-5 shows strong gains with SG (e.g., 10.78 on TUD-L, 5.80 on T-LESS, 4.09 on LM-O). On YCB-V, tuning τ_+ from 0.6 to 0.8 improves PQ_r from 1.89 to 3.12, highlighting dataset sensitivity and the benefit of mild hyperparameter tuning.

3. Details of the Results Reported in Table 4.1

Complex MPG + SG + CW. Contrary to the trend toward complexity, we show simple designs can be competitive and practical. For example, we built a more sophisticated MPG with intra/inter-token scoring to generate $|C_k| = 10$ object-specific prototypes plus a common-token prototype, followed by SLERP-based masked token averaging. Despite the added complexity, gains were marginal with extra time, compared to our single-prototype MPG.

Table-A III-1 Detailed values of all methods (SAM6D, SAM6D^f, MPG, MPG + SG, MPG + SG + CW, and Complex MPG + SG + CW) for the evaluation reported in Table 4.1.

Labels **L**, **M**, and **S** in the evaluation metrics stand for **Large**, **Medium**, and **Small**, respectively. Refer to (Hodan *et al.*, 2024) for details about the evaluation metrics. *: Zero values mean that there are no detections for that specific dataset and size

Datasets	Evaluation Metrics												
	AP	AP50	AP75	AP-L	AP-M	AP-S	AR1	AR10	AR100	AR-L	AR-M	AR-S	Time (s)
SAM6D													
HB	0.591	0.761	0.675	0.712	0.601	0.003	0.627	0.639	0.639	0.777	0.642	0.034	2.582
IC-BIN	0.362	0.663	0.376	0.504	0.404	0.004	0.091	0.418	0.456	0.645	0.484	0.021	2.394
ITODD	0.321	0.539	0.335	0.348	0.163	0*	0.175	0.473	0.478	0.502	0.260	0*	2.425
LM-O	0.463	0.797	0.501	0.457	0.512	0.079	0.525	0.543	0.543	0.618	0.590	0.115	2.181
T-LESS	0.443	0.564	0.510	0.472	0.373	0.009	0.480	0.543	0.543	0.627	0.439	0.056	2.328
TUD-L	0.575	0.878	0.687	0.670	0.571	0.550	0.626	0.657	0.657	0.712	0.646	0.750	1.992
YCB-V	0.604	0.851	0.688	0.646	0.499	0.000	0.668	0.673	0.673	0.718	0.540	0.000	2.358
SAM6D ^f													
HB	0.543	0.735	0.640	0.723	0.550	0.009	0.574	0.585	0.585	0.811	0.584	0.017	0.208
IC-BIN	0.301	0.536	0.327	0.496	0.325	0.002	0.091	0.337	0.364	0.589	0.383	0.006	0.240
ITODD	0.314	0.534	0.324	0.322	0.123	0*	0.168	0.449	0.452	0.457	0.176	0*	0.122
LM-O	0.424	0.752	0.440	0.473	0.475	0.044	0.477	0.493	0.493	0.589	0.546	0.055	0.305
T-LESS	0.416	0.538	0.490	0.469	0.346	0.006	0.451	0.512	0.512	0.606	0.403	0.029	0.262
TUD-L	0.517	0.857	0.594	0.604	0.516	0.545	0.571	0.581	0.581	0.646	0.570	0.650	0.158
YCB-V	0.619	0.863	0.718	0.660	0.505	0.000	0.678	0.684	0.684	0.728	0.541	0.000	0.175
MPG													
HB	0.579	0.741	0.657	0.732	0.588	0.004	0.612	0.620	0.620	0.798	0.624	0.040	1.518
IC-BIN	0.369	0.664	0.374	0.528	0.402	0.007	0.091	0.418	0.461	0.651	0.486	0.021	1.438
ITODD	0.334	0.545	0.352	0.361	0.183	0*	0.181	0.496	0.501	0.527	0.279	0*	2.168
LM-O	0.460	0.787	0.500	0.498	0.514	0.076	0.517	0.537	0.537	0.636	0.591	0.122	1.371
T-LESS	0.458	0.562	0.519	0.497	0.389	0.012	0.492	0.563	0.563	0.640	0.461	0.066	1.116
TUD-L	0.587	0.885	0.707	0.672	0.585	0.549	0.637	0.661	0.661	0.711	0.650	0.750	1.115
YCB-V	0.607	0.845	0.697	0.653	0.481	0.000	0.668	0.674	0.674	0.726	0.517	0.000	1.394
MPG + SG													
HB	0.570	0.733	0.646	0.697	0.580	0.004	0.603	0.610	0.610	0.774	0.615	0.040	0.724
IC-BIN	0.370	0.665	0.374	0.530	0.402	0.004	0.091	0.420	0.457	0.652	0.481	0.018	0.786
ITODD	0.336	0.546	0.357	0.360	0.198	0*	0.182	0.492	0.496	0.522	0.285	0*	0.900
LM-O	0.462	0.795	0.503	0.492	0.517	0.077	0.523	0.537	0.537	0.626	0.592	0.122	0.880
T-LESS	0.454	0.557	0.514	0.491	0.385	0.021	0.488	0.558	0.558	0.634	0.458	0.066	0.575
TUD-L	0.590	0.892	0.709	0.672	0.588	0.593	0.644	0.662	0.662	0.711	0.651	0.750	0.562
YCB-V	0.607	0.843	0.697	0.653	0.488	0.000	0.668	0.673	0.673	0.725	0.519	0.000	1.046
MPG + SG + CW													
HB	0.576	0.732	0.657	0.701	0.585	0.004	0.604	0.613	0.613	0.764	0.617	0.040	0.710
IC-BIN	0.379	0.667	0.394	0.567	0.409	0.003	0.096	0.423	0.459	0.669	0.482	0.017	0.773
ITODD	0.352	0.559	0.382	0.376	0.201	0*	0.182	0.500	0.504	0.532	0.278	0*	0.885
LM-O	0.471	0.803	0.515	0.514	0.526	0.077	0.528	0.541	0.541	0.633	0.596	0.122	0.862
T-LESS	0.462	0.565	0.523	0.497	0.392	0.023	0.493	0.559	0.559	0.637	0.460	0.066	0.568
TUD-L	0.586	0.877	0.713	0.680	0.583	0.750	0.643	0.668	0.668	0.711	0.658	0.750	0.550
YCB-V	0.618	0.850	0.709	0.665	0.503	0.000	0.671	0.676	0.676	0.726	0.526	0.000	1.040
Complex MPG + SG + CW													
HB	0.596	0.760	0.679	0.765	0.603	0.004	0.629	0.638	0.638	0.836	0.640	0.040	1.365
IC-BIN	0.366	0.662	0.367	0.523	0.399	0.006	0.093	0.415	0.458	0.644	0.483	0.019	1.454
ITODD	0.336	0.546	0.355	0.364	0.190	0*	0.182	0.498	0.503	0.530	0.290	0*	1.454
LM-O	0.460	0.787	0.499	0.490	0.514	0.081	0.518	0.533	0.533	0.642	0.587	0.118	0.826
T-LESS	0.458	0.563	0.519	0.497	0.391	0.012	0.493	0.564	0.564	0.640	0.463	0.066	1.162
TUD-L	0.584	0.883	0.698	0.672	0.581	0.505	0.632	0.659	0.659	0.712	0.648	0.750	1.369
YCB-V	0.610	0.851	0.700	0.655	0.481	0.000	0.670	0.676	0.676	0.727	0.517	0.000	1.300

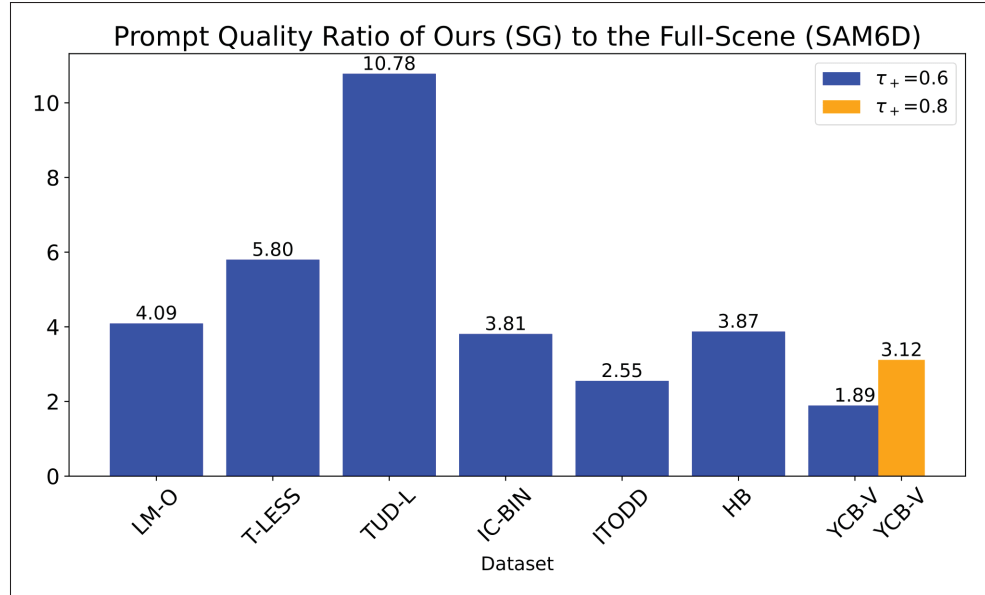


Figure-A III-5 Prompt Quality Ratio of SG vs. full-scene (SAM6D) across BOP datasets. Blue: $\tau_+ = 0.6$ (default). Orange: $\tau_+ = 0.8$ (dataset-tuned)

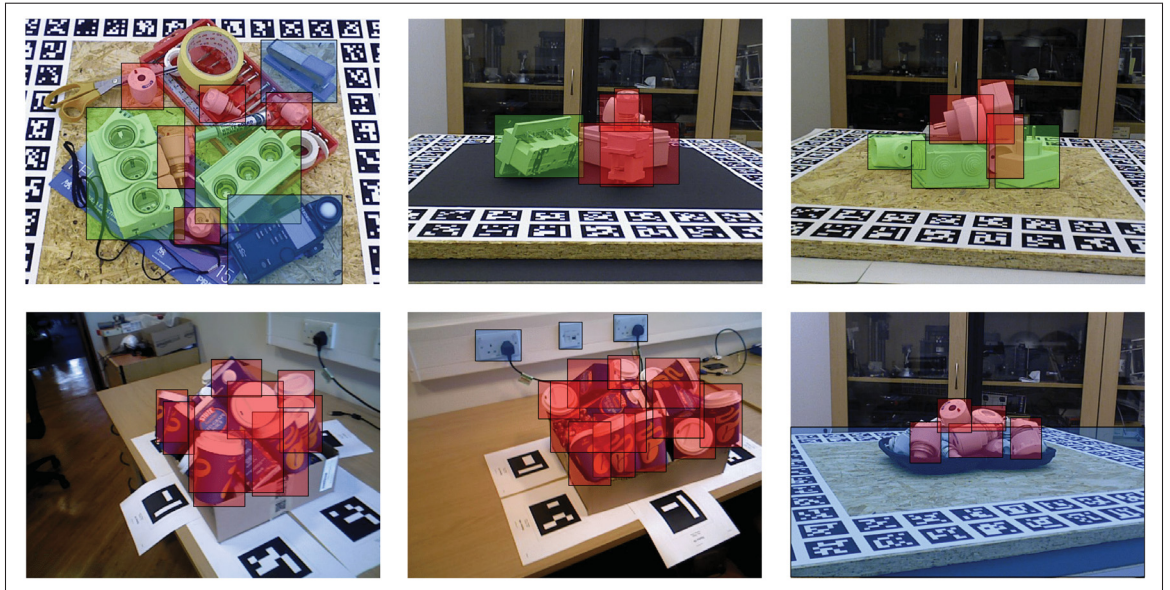


Figure-A III-6 Grounding DINO (Liu *et al.*, 2023) on BOP scenes using the prompt of (Lu *et al.*, 2024). **Green**: correct detections; **Blue**: background as object; **Red**: missed objects. Large boxes covering multiple objects cause misses since SAM returns only three masks per prompt, one often background

BIBLIOGRAPHY

- Adolfo Vargas Hakim, G., Osowiechi, D., Noori, M., Cheraghalikhani, M., Bahri, A., Yazdanpanah, M., Ben Ayed, I. & Desrosiers, C. (2024). CLIPArTT: Light-weight Adaptation of CLIP to New Domains at Test Time. *arXiv e-prints*, arXiv-2405.
- Arbel, M., Korba, A., Salim, A. & Gretton, A. (2019). Maximum mean discrepancy gradient flow. *Advances in Neural Information Processing Systems*, 32.
- Ba, J. L., Kiros, J. R. & Hinton, G. E. (2016). Layer normalization. *arXiv preprint arXiv:1607.06450*.
- Bahri, A., Yazdanpanah, M., Noori, M., Cheraghalikhani, M., Hakim, G. A. V., Osowiechi, D., Beizae, F., Ayed, I. B. & Desrosiers, C. (2024a). GeoMask3D: Geometrically Informed Mask Selection for Self-Supervised Point Cloud Learning in 3D. *arXiv preprint arXiv:2405.12419*.
- Bahri, A., Yazdanpanah, M., Noori, M., Dastani, S., Cheraghalikhani, M., Osowiech, D., Beizae, F., Ayed, I. B., Desrosiers, C. et al. (2024b). Test-Time Adaptation in Point Clouds: Leveraging Sampling Variation with Weight Averaging. *arXiv preprint arXiv:2411.01116*.
- Bahri, A., Yazdanpanah, M., Dastani, S., Noori, M., Hakim, G. A. V., Osowiechi, D., Beizae, F., Ayed, I. B. & Desrosiers, C. (2025a). SMART-PC: Skeletal Model Adaptation for Robust Test-Time Training in Point Clouds. *Proceedings of the 42nd International Conference on Machine Learning (ICML 2025)*.
- Bahri, A., Yazdanpanah, M., Noori, M., Dastani, S., Cheraghalikhani, M., Hakim, G. A. V., Osowiechi, D., Beizae, F., Ben Ayed, I. & Desrosiers, C. (2025b). Spectral informed mamba for robust point cloud processing. *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 11799–11809.
- Boudiaf, M., Mueller, R., Ben Ayed, I. & Bertinetto, L. (2022). Parameter-free online test-time adaptation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8344–8353.
- Brachmann, E., Krull, A., Michel, F., Gumhold, S., Shotton, J. & Rother, C. (2014). Learning 6d object pose estimation using 3d object coordinates. *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part II 13*, pp. 536–551.

- Cha, J., Chun, S., Lee, K., Cho, H.-C., Park, S., Lee, Y. & Park, S. (2021). SWAD: Domain generalization by seeking flat minima. *Advances in Neural Information Processing Systems*, 34, 22405–22418.
- Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H. et al. (2015). Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*.
- Chen, G., Wang, M., Yang, Y., Yu, K., Yuan, L. & Yue, Y. (2024a). Pointgpt: Auto-regressively generative pre-training from point clouds. *Advances in Neural Information Processing Systems*, 36.
- Chen, J., Zhou, Z., Sun, M., Zhao, R., Wu, L., Bao, T. & He, Z. (2024b). ZeroPose: CAD-Prompted Zero-shot Object 6D Pose Estimation in Cluttered Scenes. *IEEE Transactions on Circuits and Systems for Video Technology*.
- Chen, T., Kornblith, S., Norouzi, M. & Hinton, G. (2020). A simple framework for contrastive learning of visual representations. *International conference on machine learning*, pp. 1597–1607.
- Chen, X. & He, K. (2021). Exploring simple siamese representation learning. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 15750–15758.
- Dastani, S., Bahri, A., Yazdanpanah, M., Noori, M., Osowiechi, D., Hakim, G. A. V., Beizaee, F., Cheraghalikhani, M., Mondal, A. K., Lombaert, H. et al. (2025). Spectral State Space Model for Rotation-Invariant Visual Representation Learning. *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 23881–23890.
- De, S. & Smith, S. (2020). Batch normalization biases residual blocks towards the identity function in deep networks. *Advances in Neural Information Processing Systems*, 33, 19964–19975.
- Devlin, J. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
- Dong, R., Qi, Z., Zhang, L., Zhang, J., Sun, J., Ge, Z., Yi, L. & Ma, K. (2022). Autoencoders as cross-modal teachers: Can pretrained 2D image transformers help 3D representation learning? *arXiv preprint arXiv:2212.08320*.
- Doumanoglou, A., Kouskouridas, R., Malassiotis, S. & Kim, T.-K. (2016). Recovering 6D object pose and predicting next-best-view in the crowd. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3583–3592.

- Drost, B., Ulrich, M., Bergmann, P., Hartinger, P. & Steger, C. (2017). Introducing mvtec itodd-a dataset for 3d object recognition in industry. *Proceedings of the IEEE international conference on computer vision workshops*, pp. 2200–2208.
- Felzenszwalb, P. F., Girshick, R. B., McAllester, D. & Ramanan, D. (2009). Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9), 1627–1645.
- Gandelsman, Y., Sun, Y., Chen, X. & Efros, A. (2022). Test-time training with masked autoencoders. *Advances in Neural Information Processing Systems*, 35, 29374–29385.
- Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., March, M. & Lempitsky, V. (2016). Domain-adversarial training of neural networks. *Journal of machine learning research*, 17(59), 1–35.
- Gao, J., Zhang, J., Liu, X., Darrell, T., Shelhamer, E. & Wang, D. (2023). Back to the source: Diffusion-driven adaptation to test-time corruption. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11786–11796.
- Girshick, R. (2015). Fast r-cnn. *arXiv preprint arXiv:1504.08083*.
- Gu, A. & Dao, T. (2023). Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*.
- Gupta, A., Ozdemir, A. & Anumanchipalli, G. (2024). Geometric Interpretation of Layer Normalization and a Comparative Analysis with RMSNorm. *arXiv preprint arXiv:2409.12951*.
- Hakim, G. A. V., Osowiechi, D., Noori, M., Cheraghalikhani, M., Bahri, A., Yazdanpanah, M., Ayed, I. B. & Desrosiers, C. (2025). Clipartt: Adaptation of clip to new domains at test time. *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 7092–7101.
- Hatem, A., Qian, Y. & Wang, Y. (2023a). Point-tta: Test-time adaptation for point cloud registration using multitask meta-auxiliary learning. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16494–16504.
- Hatem, A., Qian, Y. & Wang, Y. (2023b). Test-time adaptation for point cloud upsampling using meta-learning. *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1284–1291.
- He, K., Gkioxari, G., Dollár, P. & Girshick, R. (2017). Mask r-cnn. *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969.

- Hodan, T., Haluza, P., Obdržálek, Š., Matas, J., Lourakis, M. & Zabulis, X. (2017). T-LESS: An RGB-D dataset for 6D pose estimation of texture-less objects. *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 880–888.
- Hodan, T., Sundermeyer, M., Labbe, Y., Nguyen, V. N., Wang, G., Brachmann, E., Drost, B., Lepetit, V., Rother, C. & Matas, J. (2024). BOP Challenge 2023 on Detection Segmentation and Pose Estimation of Seen and Unseen Rigid Objects. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5610–5619.
- Huang, S., Xie, Y., Zhu, S.-C. & Zhu, Y. (2021). Spatio-temporal self-supervised representation learning for 3D point clouds. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6535–6545.
- Huang, X. & Belongie, S. (2017). Arbitrary style transfer in real-time with adaptive instance normalization. *Proceedings of the IEEE international conference on computer vision*, pp. 1501–1510.
- Ioffe, S. & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *International conference on machine learning*, pp. 448–456.
- Iwasawa, Y. & Matsuo, Y. (2021). Test-time classifier adjustment module for model-agnostic domain generalization. *Advances in Neural Information Processing Systems*, 34, 2427–2440.
- Izmailov, P., Podoprikin, D., Garipov, T., Vetrov, D. & Wilson, A. G. (2018). Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*.
- Kaskman, R., Zakharov, S., Shugurov, I. & Ilic, S. (2019). Homebreweddb: Rgb-d dataset for 6d pose estimation of 3d objects. *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pp. 0–0.
- Kim, K., Laskin, M., Mordatch, I. & Pathak, D. (2021). How to adapt your large-scale vision-and-language model.
- Kim, S., Shen, S., Thorsley, D., Gholami, A., Kwon, W., Hassoun, J. & Keutzer, K. (2022). Learned token pruning for transformers. *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 784–794.
- Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A. C., Lo, W.-Y. et al. (2023). Segment anything. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4015–4026.

- Kuo, W., Angelova, A., Malik, J. & Lin, T.-Y. (2019). Shapemask: Learning to segment novel objects by refining shape priors. *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9207–9216.
- Lee, D.-H. et al. (2013). Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. *Workshop on challenges in representation learning, ICML*, 3(2), 896.
- Li, X., Jiang, M., Zhang, X., Kamp, M. & Dou, Q. (2021). Fedbn: Federated learning on non-iid features via local batch normalization. *arXiv preprint arXiv:2102.07623*.
- Li, Y., Wang, N., Shi, J., Liu, J. & Hou, X. (2016). Revisiting batch normalization for practical domain adaptation. *arXiv preprint arXiv:1603.04779*.
- Liang, D., Zhou, X., Xu, W., Zhu, X., Zou, Z., Ye, X., Tan, X. & Bai, X. (2024). Pointmamba: A simple state space model for point cloud analysis. *Advances in neural information processing systems*, 37, 32653–32677.
- Liang, J., Hu, D. & Feng, J. (2020). Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. *International conference on machine learning*, pp. 6028–6039.
- Lim, H., Kim, B., Choo, J. & Choi, S. (2023). TTN: A domain-shift aware batch normalization in test-time adaptation. *arXiv preprint arXiv:2302.05155*.
- Lin, J., Liu, L., Lu, D. & Jia, K. (2024). SAM-6d: Segment anything model meets zero-shot 6d object pose estimation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 27906–27916.
- Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P. & Zitnick, C. L. (2014). Microsoft coco: Common objects in context. *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part V 13*, pp. 740–755.
- Liu, H., Cai, M. & Lee, Y. J. (2022). Masked discrimination for self-supervised learning on point clouds. *European Conference on Computer Vision*, pp. 657–675.
- Liu, S., Zeng, Z., Ren, T., Li, F., Zhang, H., Yang, J., Li, C., Yang, J., Su, H., Zhu, J. et al. (2023). Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*.

- Liu, Y., Tian, Y., Zhao, Y., Yu, H., Xie, L., Wang, Y., Ye, Q., Jiao, J. & Liu, Y. (2024). Vmamba: Visual state space model. *Advances in neural information processing systems*, 37, 103031–103063.
- Liu, Y., Kothari, P., Van Delft, B., Bellot-Gurlet, B., Mordan, T. & Alahi, A. (2021). Ttt++: When does self-supervised test-time training fail or thrive? *Advances in Neural Information Processing Systems*, 34, 21808–21820.
- Lu, Y., Guo, Y., Ruozzi, N., Xiang, Y. et al. (2024). Adapting Pre-Trained Vision Models for Novel Instance Detection and Segmentation. *arXiv preprint arXiv:2405.17859*.
- Lubana, E. S., Dick, R. & Tanaka, H. (2021). Beyond batchnorm: Towards a unified understanding of normalization in deep learning. *Advances in Neural Information Processing Systems*, 34, 4778–4791.
- Luo, P., Zhang, R., Ren, J., Peng, Z. & Li, J. (2019). Switchable normalization for learning-to-normalize deep representation. *IEEE transactions on pattern analysis and machine intelligence*, 43(2), 712–728.
- Lyu, K., Li, Z. & Arora, S. (2022). Understanding the generalization benefit of normalization layers: Sharpness reduction. *Advances in Neural Information Processing Systems*, 35, 34689–34708.
- Mirza, M. J., Micorek, J., Possegger, H. & Bischof, H. (2022). The norm must go on: Dynamic unsupervised domain adaptation by normalization. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 14765–14775.
- Mirza, M. J., Shin, I., Lin, W., Schriebl, A., Sun, K., Choe, J., Kozinski, M., Possegger, H., Kweon, I. S., Yoon, K.-J. et al. (2023). Mate: Masked autoencoders are online 3d test-time learners. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16709–16718.
- Mounsaveng, S., Chiaroni, F., Boudiaf, M., Pedersoli, M. & Ben Ayed, I. (2024). Bag of tricks for fully test-time adaptation. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1936–1945.
- Nado, Z., Padhy, S., Sculley, D., D’Amour, A., Lakshminarayanan, B. & Snoek, J. (2020). Evaluating prediction-time batch normalization for robustness under covariate shift. *arXiv preprint arXiv:2006.10963*.
- Nguyen, V. N., Groueix, T., Ponimatkin, G., Lepetit, V. & Hodan, T. (2023). Cnos: A strong baseline for cad-based novel object segmentation. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2134–2140.

- Noori, M., Osowiechi, D., Hakim, G. A. V., Bahri, A., Yazdanpanah, M., Dastani, S., Beizae, F., Ayed, I. B. & Desrosiers, C. (2025). Test-Time Adaptation of Vision-Language Models for Open-Vocabulary Semantic Segmentation. *arXiv preprint arXiv:2505.21844*.
- Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A. et al. (2023). Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*.
- Örnek, E. P., Labbé, Y., Tekin, B., Ma, L., Keskin, C., Forster, C. & Hodan, T. (2025). Foundpose: Unseen object pose estimation with foundation features. *European Conference on Computer Vision*, pp. 163–182.
- Osowiechi, D., Hakim, G. A. V., Noori, M., Cheraghalikhani, M., Ben Ayed, I. & Desrosiers, C. (2023). TTTFlow: Unsupervised test-time training with normalizing flow. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 2126–2134.
- Osowiechi, D., Hakim, G. A. V., Noori, M., Cheraghalikhani, M., Bahri, A., Yazdanpanah, M., Ben Ayed, I. & Desrosiers, C. (2024a). Nc-ttt: A noise constrastive approach for test-time training. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6078–6086.
- Osowiechi, D., Noori, M., Vargas Hakim, G., Yazdanpanah, M., Bahri, A., Cheraghalikhani, M., Dastani, S., Beizae, F., Ayed, I. & Desrosiers, C. (2024b). WATT: Weight average test time adaptation of CLIP. *Advances in neural information processing systems*, 37, 48015–48044.
- Osowiechi, D., Noori, M., Vargas Hakim, G., Yazdanpanah, M., Bahri, A., Cheraghalikhani, M., Dastani, S., Beizae, F., Ayed, I. & Desrosiers, C. (2025). WATT: Weight Average Test Time Adaptation of CLIP. *Advances in Neural Information Processing Systems*, 37, 48015–48044.
- Pang, Y., Wang, W., Tay, F. E., Liu, W., Tian, Y. & Yuan, L. (2022). Masked autoencoders for point cloud self-supervised learning. *European conference on computer vision*, pp. 604–621.
- Qi, C. R., Su, H., Mo, K. & Guibas, L. J. (2017a). PointNet: Deep learning on point sets for 3D classification and segmentation. *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 652–660.
- Qi, C. R., Yi, L., Su, H. & Guibas, L. J. (2017b). PointNet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30.

- Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J. et al. (2021). Learning transferable visual models from natural language supervision. *International conference on machine learning*, pp. 8748–8763.
- Ren, T., Liu, S., Zeng, A., Lin, J., Li, K., Cao, H., Chen, J., Huang, X., Chen, Y., Yan, F. et al. (2024). Grounded SAM: Assembling open-world models for diverse visual tasks. *arXiv preprint arXiv:2401.14159*.
- Roburin, S., de Mont-Marin, Y., Bursuc, A., Marlet, R., Perez, P. & Aubry, M. (2022). Spherical perspective on learning with normalization layers. *Neurocomputing*, 487, 66–74.
- Schneider, S., Rusak, E., Eck, L., Bringmann, O., Brendel, W. & Bethge, M. (2020). Improving robustness against common corruptions by covariate shift adaptation. *Advances in neural information processing systems*, 33, 11539–11551.
- Shim, H., Kim, C. & Yang, E. (2024). Cloudfixer: Test-time adaptation for 3d point clouds via diffusion-guided geometric transformation. *European Conference on Computer Vision*, pp. 454–471.
- Shin, I., Tsai, Y.-H., Zhuang, B., Schuler, S., Liu, B., Garg, S., Kweon, I. S. & Yoon, K.-J. (2022). Mm-tta: multi-modal test-time adaptation for 3d semantic segmentation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16928–16937.
- Socher, R., Karpathy, A., Le, Q. V., Manning, C. D. & Ng, A. Y. (2014). Grounded compositional semantics for finding and describing images with sentences. *Transactions of the Association for Computational Linguistics*, 2, 207–218.
- Sun, J., Zhang, Q., Kailkhura, B., Yu, Z., Xiao, C. & Mao, Z. M. (2022). Benchmarking robustness of 3d point cloud recognition against common corruptions. *arXiv preprint arXiv:2201.12296*.
- Sun, Y., Wang, X., Liu, Z., Miller, J., Efros, A. & Hardt, M. (2020). Test-time training with self-supervision for generalization under distribution shifts. *International conference on machine learning*, pp. 9229–9248.
- Sundermeyer, M., Hodaň, T., Labbe, Y., Wang, G., Brachmann, E., Drost, B., Rother, C. & Matas, J. (2023). Bop challenge 2022 on detection, segmentation and pose estimation of specific rigid objects. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2785–2794.

- Tang, Q., Zhang, B., Liu, J., Liu, F. & Liu, Y. (2023). Dynamic token pruning in plain vision transformers for semantic segmentation. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 777–786.
- Tran, T. A., Nguyen, D. M., Tran, H.-C., Barz, M., Doan, K. D., Wattenhofer, R., Vien, N. A., Niepert, M., Sonntag, D. & Swoboda, P. (2025). How Many Tokens Do 3D Point Cloud Transformer Architectures Really Need? *arXiv preprint arXiv:2511.05449*.
- Uy, M. A., Pham, Q.-H., Hua, B.-S., Nguyen, T. & Yeung, S.-K. (2019). Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1588–1597.
- Vaswani, A. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*.
- Wang, D., Shelhamer, E., Liu, S., Olshausen, B. & Darrell, T. (2020). Tent: Fully test-time adaptation by entropy minimization. *arXiv preprint arXiv:2006.10726*.
- Wang, H., Sridhar, S., Huang, J., Valentin, J., Song, S. & Guibas, L. J. (2019a). Normalized object coordinate space for category-level 6d object pose and size estimation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2642–2651.
- Wang, H., Dedhia, B. & Jha, N. K. (2024a). Zero-TPrune: Zero-shot token pruning through leveraging of the attention graph in pre-trained transformers. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16070–16079.
- Wang, H., Ma, S., Dong, L., Huang, S., Zhang, D. & Wei, F. (2024b). Deepnet: Scaling transformers to 1,000 layers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(10), 6761–6774.
- Wang, Y., Cheraghian, A., Hayder, Z., Hong, J., Ramasinghe, S., Rahman, S., Ahmedt-Aristizabal, D., Li, X., Petersson, L. & Harandi, M. (2024c). Backpropagation-free network for 3d test-time adaptation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 23231–23241.
- Wang, Y., Hong, J., Cheraghian, A., Rahman, S., Ahmedt-Aristizabal, D., Petersson, L. & Harandi, M. (2024d). Continual test-time domain adaptation via dynamic sample selection. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1701–1710.

- Wang, Y., Sun, Y., Liu, Z., Sarma, S. E., Bronstein, M. M. & Solomon, J. M. (2019b). Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (tog)*, 38(5), 1–12.
- Wang, Z., Luo, Y., Zheng, L., Chen, Z., Wang, S. & Huang, Z. (2025). In search of lost online test-time adaptation: A survey. *International Journal of Computer Vision*, 133(3), 1106–1139.
- Wen, B., Yang, W., Kautz, J. & Birchfield, S. (2024). Foundationpose: Unified 6d pose estimation and tracking of novel objects. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 17868–17879.
- Wu, X., Zeng, F., Wang, X. & Chen, X. (2023). Ppt: Token pruning and pooling for efficient vision transformers. *arXiv preprint arXiv:2310.01812*.
- Xiang, Y., Schmidt, T., Narayanan, V. & Fox, D. (2017). Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *arXiv preprint arXiv:1711.00199*.
- Xiong, R., Yang, Y., He, D., Zheng, K., Zheng, S., Xing, C., Zhang, H., Lan, Y., Wang, L. & Liu, T. (2020). On layer normalization in the transformer architecture. *International conference on machine learning*, pp. 10524–10533.
- Xu, J., Sun, X., Zhang, Z., Zhao, G. & Lin, J. (2019). Understanding and improving layer normalization. *Advances in neural information processing systems*, 32.
- Yan, S., Yang, Z., Li, H., Song, C., Guan, L., Kang, H., Hua, G. & Huang, Q. (2023). Implicit autoencoder for point-cloud self-supervised representation learning. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 14530–14542.
- Yazdanpanah, M. & Moradi, P. (2022). Visual domain bridge: A source-free domain adaptation for cross-domain few-shot learning. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2868–2877.
- Yazdanpanah, M., Rahman, A. A., Chaudhary, M., Desrosiers, C., Havaei, M., Belilovsky, E. & Kahou, S. E. (2022). Revisiting learnable affines for batch norm in few-shot transfer learning. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9109–9118.
- Yazdanpanah, M., Bahri, A., Noori, M., Dastani, S., Hakim, G. A. V., Osowiechi, D., Ayed, I. B. & Desrosiers, C. (2025). Purge-Gate: Backpropagation-Free Test-Time Adaptation for Point Clouds Classification via Token Purging. *arXiv preprint arXiv:2509.09785*.

- Yeo, T., Kar, O. F., Sodagar, Z. & Zamir, A. (2023). Rapid network adaptation: Learning to adapt neural networks using test-time feedback. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4674–4687.
- Yu, X., Tang, L., Rao, Y., Huang, T., Zhou, J. & Lu, J. (2022). Point-BERT: Pre-training 3D point cloud transformers with masked point modeling. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 19313–19322.
- Zbontar, J., Jing, L., Misra, I., LeCun, Y. & Deny, S. (2021). Barlow twins: Self-supervised learning via redundancy reduction. *International conference on machine learning*, pp. 12310–12320.
- Zhang, B. & Sennrich, R. (2019). Root mean square layer normalization. *Advances in neural information processing systems*, 32.
- Zhang, H., Li, F., Liu, S., Zhang, L., Su, H., Zhu, J., Ni, L. M. & Shum, H.-Y. (2022a). Dino: Detr with improved denoising anchor boxes for end-to-end object detection. *arXiv preprint arXiv:2203.03605*.
- Zhang, J., Qi, L., Shi, Y. & Gao, Y. (2023a). Domainadaptor: A novel approach to test-time adaptation. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 18971–18981.
- Zhang, J., Liu, S., Song, J., Zhu, T., Xu, Z. & Song, M. (2024). Lookaround Optimizer: k steps around, 1 step average. *Advances in Neural Information Processing Systems*, 36.
- Zhang, M., Levine, S. & Finn, C. (2022b). Memo: Test time robustness via adaptation and augmentation. *Advances in neural information processing systems*, 35, 38629–38642.
- Zhang, R., Guo, Z., Gao, P., Fang, R., Zhao, B., Wang, D., Qiao, Y. & Li, H. (2022c). Point-M2AE: Multi-scale masked autoencoders for hierarchical point cloud pre-training. *Advances in neural information processing systems*, 35, 27061–27074.
- Zhang, R., Wang, L., Qiao, Y., Gao, P. & Li, H. (2023b). Learning 3D representations from 2D pre-trained models via image-to-point masked autoencoders. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21769–21780.
- Zhao, H., Liu, Y., Alahi, A. & Lin, T. (2023a). On pitfalls of test-time adaptation. *arXiv preprint arXiv:2306.03536*.
- Zhao, X., Ding, W., An, Y., Du, Y., Yu, T., Li, M., Tang, M. & Wang, J. (2023b). Fast segment anything. *arXiv preprint arXiv:2306.12156*.

- Zhu, C. & Chen, L. (2024). A survey on open-vocabulary detection and segmentation: Past, present, and future. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- Ziaee, A. & Çano, E. (2022). Batch Layer Normalization A new normalization layer for CNNs and RNNs. *Proceedings of the 6th International Conference on Advances in Artificial Intelligence*, pp. 40–49.
- Zou, Y., Yi, S., Li, Y. & Li, R. (2024a). A Closer Look at the CLS Token for Cross-Domain Few-Shot Learning. *Advances in Neural Information Processing Systems*, 37, 85523–85545.
- Zou, Y., Yi, S., Li, Y. & Li, R. (2024b). A Closer Look at the CLS Token for Cross-Domain Few-Shot Learning. *NeurIPS*, 37, 85523–85545.