Target-Guided Adversarial Point Cloud Transformer Towards Recognition Against Real-world Corruptions

Jie Wang[1], Tingfa Xu[1,†], Lihe Ding[2], Jianan Li[1,†]

[1]Beijing Institute of Technology   [2]The Chinese University of Hong Kong

https://github.com/Roywangj/APCT

Abstract

Achieving robust 3D perception in the face of corrupted data presents an challenging hurdle within 3D vision research. Contemporary transformer-based point cloud recognition models, albeit advanced, tend to overfit to specific patterns, consequently undermining their robustness against corruption. In this work, we introduce the Target-Guided **A**dversarial **P**oint **C**loud **T**ransformer, termed **APCT**, a novel architecture designed to augment global structure capture through an adversarial feature erasing mechanism predicated on patterns discerned at each step during training. Specifically, APCT integrates an Adversarial Significance Identifier and a Target-guided Promptor. The Adversarial Significance Identifier, is tasked with discerning token significance by integrating global contextual analysis, utilizing a structural salience index algorithm alongside an auxiliary supervisory mechanism. The Target-guided Promptor, is responsible for accentuating the propensity for token discard within the self-attention mechanism, utilizing the value derived above, consequently directing the model attention towards alternative segments in subsequent stages. By iteratively applying this strategy in multiple steps during training, the network progressively identifies and integrates an expanded array of object-associated patterns. Extensive experiments demonstrate that our method achieves state-of-the-art results on multiple corruption benchmarks.

1 Introduction

3D point cloud recognition has garnered significant interest owing to its promising implications for robotics and autonomous driving. Prevailing techniques [11, 12, 20, 13] have been predominantly designed and evaluated on clean data [22, 1], overlooking the extensive corruptions present in real-world scenarios arising from sensor inaccuracies and physical constraints and leading to suboptimal performance when models are exposed to such conditions. Therefore, enhancing the resilience of point cloud models against real-world corruption emerges as a paramount yet daunting task.

Due to their elevated performance, existing transformer-based models [4, 15] have emerged as the mainstream choice. Despite their effectiveness on clean datasets, these methods generally falter when faced with corrupted data. Upon closer examination, we discerned that prevailing models have a propensity to overfit to specific patterns (Fig. 1a). Such patterns can degrade in the presence of real-world corruptions, undermining the model reliability. Relying solely on these localized patterns for predictions can be fragile, especially with corrupted data. Thus, we theorize that if the model is encouraged to extract features from a broader region of the object during training, gathering more diverse perception cues, it would be more resilient, as proved in Fig. 1b. This is because, even if some local patterns are compromised in corrupted data, the model could still source information from other intact areas to make accurate predictions.

To address the aforementioned limitation, we present a novel adversarial representation learning method, named Target-Guided Adversarial Point Cloud Transformer(APCT), for robust 3D perception.

---

†Correspondence to: Jianan Li and Tingfa Xu.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

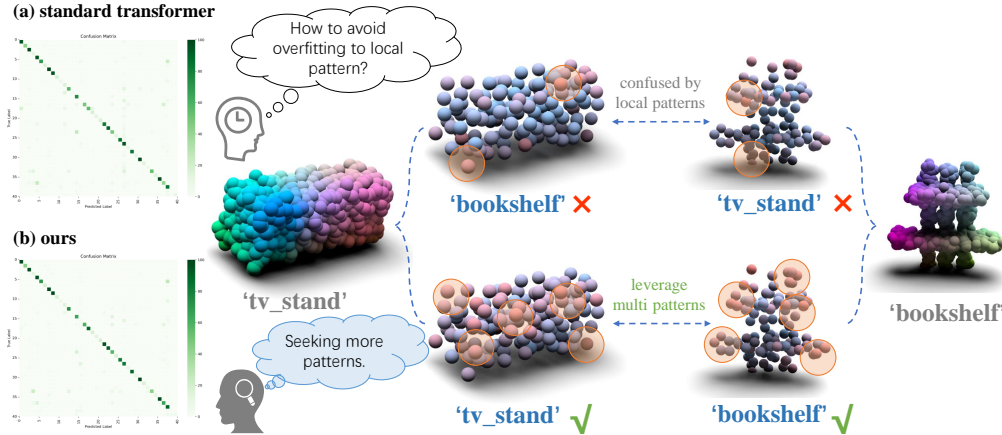



Figure 1: **Overall motivation**. We advocate for the model to broaden its attention to diverse patterns, mitigating the tendency to overfit to localized patterns. The left segment of the figure contrasts the confusion matrices of the standard transformer with our approach. The right portion showcases the performances of both the standard transformer and our methodology when confronted with objects exhibiting similar local patterns. Tokens with high / low contributions to classification are in red / blue, respectively. Standard transformer tends to overfit to localized patterns. While our method, by modulating tokens with significant contributions, enables the model to garner features from a varied spectrum of target segments, thereby ensuring greater robustness.

This method adversarially weakens the dominant patterns and gathers more sub-important perception cues during training on clean samples. By progressive pattern excavation, the proposed approach prompts the model to delve into objects and acquire a broader range of patterns, consequently elevating the robustness of the model.

Specifically, we first partition and extract local geometries from the point cloud by a mini-PointNet [11], resulting in a subset of tokens that encode these features. The tokens are then fed into stacked transformer blocks with two core modules, namely the Adversarial Significance Identifier and the Target-guided Promptor. The former module utilizes a dominant feature index mechanism along with an auxiliary supervision, to discern significance of all tokens, thus integrating global contextual analysis. Simultaneously, it signs a proportion of dominant tokens that are significant for perception in the current phase. Subsequently, the Target-guided Promptor, increases the dropping likelihood of token signed above during the self-attention process, thereby compelling the network to focus on less dominant tokens and to extract perceptive clues from alternative patterns. By iteratively engaging in such an adversarial process, the network gradually excavates and assimilates an extended array of patterns from objects (in Fig. 2a), thus making precise predictions against corruption.

We have extensively validated the effectiveness of our proposed method on multiple benchmarks, including ModelNet-C [15] and the more challenging ScanObjectNN-C [19]. The results significantly underscore improvements in robustness and establish state-of-the-art performance on these datasets. In addition, our algorithm demonstrates pronounced generalization capabilities in downstream tasks, as evidenced by its adeptness in shape segmentation under corruptions in ShapeNet-C [14]. These findings collectively underscore the method's generality and effectiveness in enhancing the robustness of point cloud perception models. Our contributions can be summarized as follows:

- We present a new framework, APCT, which effectively improves the resilience of point cloud model against various types of corruptions by leveraging adversarial mining strategy.
- We propose a novel adversarial dropout method that can stimulate the model to embrace broader valuable patterns by adversarially lifting dropout probability of specific patterns.
- We demonstrate the effectiveness of APCT through extensive experiments on multiple point cloud benchmarks under diverse corruption scenarios.

2 Related Work

2.1 Transformer for Point Cloud Recognition

Amidst the burgeoning epoch of vision transformers, seminal works such as Point Transformer [32] and Point Cloud Transformer [4] ventured into the realm of leveraging attention mechanisms tailored

2

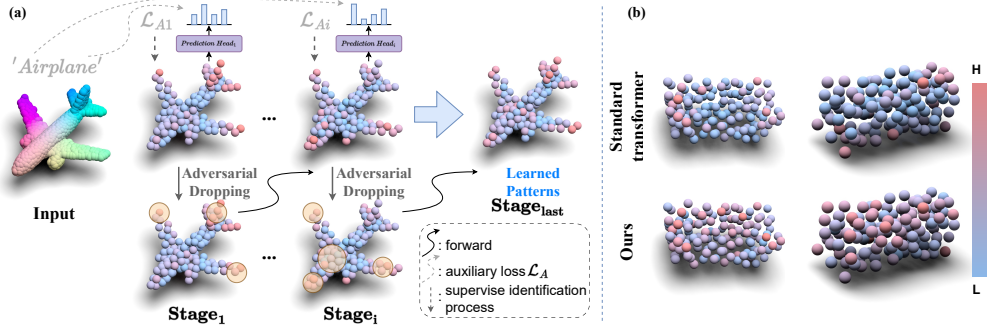



Figure 2: (a) Process of progressive adversarial dropping. The first line means token learned in each stage. (b) Visualization of token weights learned by the classifier. Compared with standard transformer, ours has an advantage in mining sample patterns.

for point cloud semantics, thus pioneering this trajectory. The Point Cloud Transformer intrinsically integrates global attention across the entirety of the point cloud, mirroring certain challenges inherent to the Vision Transformer (ViT), notably the constraints of memory bandwidth and computational overhead. Contrarily, the Point Transformer accentuates local attention, distinctly focusing on each point and its proximate neighbors, effectively mitigating the aforementioned memory impediments. Subsequent to these, the Robust Point Cloud Classifier (RPC) [15] emerged, proffering a robust framework for point cloud classification. Infusing 3D representation, k-NN, frequency grouping, and self-attention, it posits itself as a formidable baseline in the annals of point cloud robustness inquiries. While these methods demonstrate remarkable accuracy , their performance tends to deteriorate in face of real-world corruptions. In this manuscript, we introduce a novel transformer-based model specifically architected to robustly handle real-world corruptions.

2.2 Robust Learning Against Corruptions

In the realm of point cloud perception, deep learning has made significant strides [11, 12, 9, 13]. However, ensuring model robustness in the presence of corruptions remains a pivotal concern. Contemporary literature underscores the imperative of addressing point cloud corruptions [15, 14, 5, 7]. Some methodologies have hinged on preprocessing, incorporating denoising or completion strategies [34], to bolster a model's resilience against adversarial perturbations. Others have pursued augmentation pathways, spanning mix-based [7] and deformation-centric [5, 15] techniques. The advent of advanced auto-augmentations [19] has ushered in sample-adaptive data augmentations, targeting enhanced model robustness. Notwithstanding their merits, such interventions often mandate augmented computational overhead and may falter in scenarios of pronounced corruption. Another line of research aims to improve the robustness of deep learning architectures themselves. PointASNL [27] integrates adaptive sampling coupled with local-nonlocal modules, striving for heightened robustness. Concurrently, Ren *et al.* [14] unveiled an attention-based backbone, RPC, tailored to withstand corruption. Nevertheless, these methods only make improvements to the feature extractor, without concerning the nature of the corruption problem. This manuscript introduces to leverage an adversarial dropping strategy to discern pertinent features from corrupted samples.

3 Methodology

We instead devise an adversarial analysis based framework (Fig. 2), which not only learns point recognition with current vital patterns, but more essentially, it automatically discovers and emphasizes the sub-important patterns, with the auxiliary supervised pattern miner. Patterns learned in such strategy are expected to be more discriminative and robust, hence facilitating final recognition of point clouds under corruptions. At each training iteration, our algorithm comprises of two phases. In **phase 1**, we perform adversarial mining over across all patterns, based on the token significance identification process. The propose is to search for the sub-optimal patterns that contribute less to the final recognition. In **phase 2**, we leverage deterministic dropping assignments and attribute elevated dropping rates to tokens pivotal for the final perception, as an constraint to enable these sub-optimal tokens to play a increasing significant role in perceptual. Engaging in this iterative adversarial procedure allows the network to progressively unearth and assimilate a comprehensive set of object patterns, as seen in Fig. 2b, culminating in refined predictions in the face of corruption.

3

3.1 Overview Implementations

The overall implementations of Adversarial Point Cloud Transformer is shown in Fig. 3. Given an input set of point cloud $\mathcal{P} \in \mathbb{R}^{N \times 3}$, we first partition the input point cloud into an assembly of n discrete patches, and subsequently, generating pertinent tokens with C-dimensional features, $T = \{t_j | j = 1, ..., n\} \in \mathbb{R}^{n \times C}$. Taking these tokens as input for the following stages, the network initially employs the Adversarial Significance Identifier module to formulate a per-token dropping rate, assigning higher rates to tokens that are critical for classification. Subsequently, the Target-guided Promptor selectively eliminates key vectors based on the established rates. Through introducing such an adversarial progressive dropping scheme, the network gathers a diverse set of perception cues and assimilates various patterns from an expansive region of the underlying object structure, thus enhancing classifier resilience against potential corruptions.

The proposed architecture is segmented into three distinct stages, each consisting of multiple blocks. Within stages, the depth of self-attention blocks is conventionally configured to [4, 4, 4]. Each stage is further equipped with a uniformly applied Adversarial Significance Identifier and an associated complementary supervision head. The default adversarial dropping ratio, is uniformly set to [0.2, 0.2, 0.2] across all stages. Each block within stages independently incorporates a Target-guided Promptor.

3.2 Adversarial Significance Identifier

This module operates by meticulously processing the input tokens extracted from point clouds. It evaluates and distinguishs the relative significance of each token and applies constraints particularly to those tokens identified as most crucial in the context of point representation learning, thereby efficiently unveiling underlying data structures through progressively mining patterns that sub-significant yet representative and insightful. Utilizing an auxiliary supervisory process, it integrates global contextual analysis and discerns significance of all input tokens T; thus identifying the focal tokens, finally generating per-token dropout rate. To this end, it sorts tokens based on the feature channel responses via index number in auxiliary supervision process, then it establishes the mapping between tokens and the dropout rate, finally predict per-token dropout rate $M = \{m_j | j = 1, ..., n\} \in \mathbb{R}^n$. The overall workflow is shown in Fig. 4.

Focal tokens identification. In the context of point cloud analysis, let $T = \{t_j | j = 1, ..., n\} \in \mathbb{R}^{n \times C}$ represent a set comprising point cloud tokens. Initially, an essential step entails the organization of token contributions to final perception based on their associated feature channel responses, through the auxiliary supervisory process. Subsequent to this arrangement, we proceed with the identification of the most prominent tokens pertaining to each feature channel, resulting in the derivation of their respective index bank denoted as $F_{\text{topk}} = \left\{ f_{\text{topk}}^m | m = 1, ..., D \right\} \in \mathbb{R}^{k \times C}$, where k signifies the number of tokens retained per channel feature. This process can be succinctly expressed as:

$$F_{topk} = TopK(T), \tag{1}$$

where $f_{\text{topk}}^m \in \mathbb{R}^k$ enumerates values within the range $(1, N)$, indicating the token associated with the feature response under consideration. Upon the construction of the index repository F_{topk}, we perform a comprehensive aggregation of the elements within it. This process is pivotal for quantifying the occurrence frequency of each token in the ultimate perception of the model. This assists in delineating the tokens of substantial relevance. The mathematical representation is as follows:

$$M = \psi(F_{topk}), \tag{2}$$

where ψ symbolizes the function employed to compute the token significance from F_{topk} in accordance with predefined indices. Consequently, the matrix $M = \{m_j | j = 1, ..., n\} \in \mathbb{R}^n$ emerges as the importance matrix for tokens. Each element within this matrix, denoted by $M_j \in \mathbb{R}^1$, serves to quantify the representation significance of the $j - th$ token within the overall perception framework.

Supervisory token identification process. Current algorithms that optimize the network under the guidance of readily high-level labels at final layer is insufficient for guiding the selection of drop matrices in the intermediate stages of the network. This limitation may lead to the phenomena that constructed matrices do not accurately reflect the contributions of tokens, potentially compromising the efficiency of algorithm. Thus, the major question arises: *how can we optimize the selection of drop matrices to accurately represent each token's contribution?*

To rectify this, our framework innovates with an adversarial approach embedded within each stage, which cornerstone is the integration of specialized auxiliary heads, strategically positioned within the

4

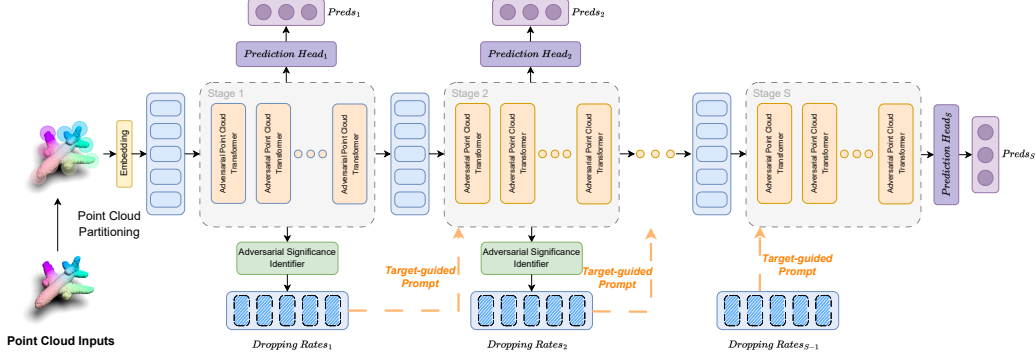



Figure 3: **Overall architecture** of our algorithm, composed of two key modules: Adversarial Significance Identifier and Target-guided Promptor. The former evaluates token significance within the context of the global perception, with the help of dominant feature indexing process from an auxiliary supervising loss that can bolster the precision of the index selection, then producing dropping rate for tokens. Subsquently, Target-guided Promptor enhances key dropout probabilities influenced by rate above, driving the model to explore auxiliary segments for pivotal information. This mechanism mitigates the propensity of the model to overfit to localized patterns.

network. Each auxiliary head is assigned the critical task of computing a unique and stage-specific loss, it identifies the patterns or tokens that the current stage predominantly focuses on and then intentionally drops these identified tokens. This process forces the network to shift its attention, thereby encouraging it to mine features from other, less emphasized regions.

The operation within these auxiliary heads begins with a dominant feature index operation applied to the tokens, which aligns with the identification methodology detailed in Eq. 1. This ensures that the drop matrices generated are a true reflection of tokens significance at every specific network stage.

Implementing these adversarial-focused auxiliary heads represents a significant leap in the optimization process. By introducing a targeted, stage-specific supervision issue at each stage, they offer a refined assessment of token contributions, a stark contrast to the broader, less specific results from the network's final layer. This precision in evaluation is pivotal to the nuanced understanding and optimization of each stage within the network, aligning token signification closely with the specific objectives of each stage in the network. The adversarial approach facilitates an augmented constraint term and plays an instrumental role in the selection of drop matrices at each stage, contributing substantially to the optimization process, encapsulated as follows:

$$\mathcal{L}_A = \frac{1}{N} \sum_{i=1}^{N} \|y^*{}_i - \hat{y}_i\|_2 \, , ` \tag{3}$$

where $\hat{y}$ and y^* denote the predicted and ground-truth labels, respectively, for the input point cloud $\mathcal{P} \in \mathbb{R}^{N \times 3}$, with $\|\cdot\|_2$ indicating the L_2 norm. This constraint term, derived from the cumulative loss computed by all the auxiliary heads, ensures a comprehensive and layered supervision across the network. It allows the Adversarial Significance Identifier to discern the contribution of each token more accurately, thereby generating more targeted and effective supervisory signals for the token dropping process. This, in turn, enhances the overall efficacy and precision of the network learning.

Derivation of per-token dropout rate. The objective of $\Phi(\cdot)$ below is to refine the raw frequency distribution of tokens, as encapsulated within the matrix M, transforming them into precise and actionable probability distribution. This transformation is critical for the effective application of dropout rates to tokens in the network. The specific functional representation is detailed below:

$$\Phi(m_j, \gamma, \alpha, \beta) = \begin{cases} \alpha & \text{if } \gamma \frac{n \cdot m_j}{\sum_{j=1}^{n} m_j} < \alpha \\ \gamma \frac{n \cdot m_j}{\sum_{j=1}^{n} m_j} & \text{if } \alpha \le \gamma \frac{n \cdot m_j}{\sum_{j=1}^{n} m_j} \le \beta \\ \beta & \text{if } \gamma \frac{n \cdot m_j}{\sum_{j=1}^{n} m_j} > \beta \end{cases} \tag{4}$$

Herein, the parameter γ represents the mapping ratio, a pivotal factor whose impact and intricacies are thoroughly examined in a later ablation study. The variable n denotes the total number of tokens under consideration. The boundary parameters, α and β, are pre-set to the values of 0.05 and 0.95, respectively. These parameters play a vital role in defining the range and sensitivity of the transformation function. The probability distribution thus generated by $\Phi(\cdot)$ is instrumental in

5

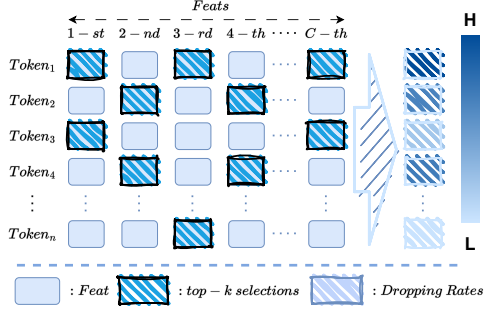



Figure 4: **Workflow of Adversarial Signifi-cance Identifier**. Squares with light/dark blue means low/high values, respectively.

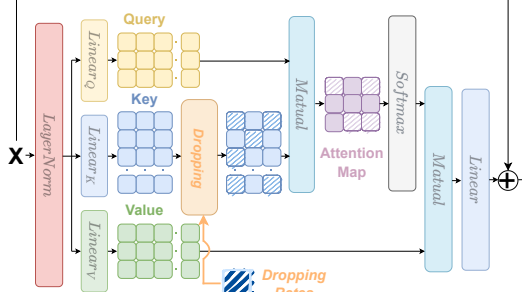



Figure 5: **The proposed Target-guided Prompting** mechanism that increases the dropout probability, based on the per-token dropping rate.

dictating the dropout strategy in the following Target-guided Promptor. This structured approach ensures that the dropout process is both targeted and efficient, enhancing the overall performance.

3.3 Target-guided Promptor

In traditional self-attention mechanism in transformers, the use of vanilla dropout techniques is prevalent, where each node in attention matrix is assigned a uniform stochastic discard probability. While this strategy aids in reducing overfitting and enhancing the generalization, it inherently overlooks the non-uniform characteristics of nodes in the matrix. The indiscriminate application of random dropout across all nodes fails to effectively penalize those with significantly higher attention scores; this oversight may result in a residual risk of overfitting to specific localized patterns. To address this shortcoming, our proposed methodology diverges from the conventional approach by independently targeting each key within the self-attention mechanism. As detailed in Fig. 5, this focused strategy efficiently penalizes keys with pronounced attention scores, thereby directly addressing the issue of overfitting to specific local patterns.

During each training epoch, this module adaptively masks a specified fraction of keys in the input key map using the matrix M. Notably, for every distinct query, a unique masked key map is synthesized, as opposed to utilizing a shared masked key map for the entire set of query vectors. Given token features defined by $T = \{t_j | j = 1, ..., n\} \in \mathbb{R}^{n \times C}$ accompanied by coordinates $\mathcal{D}$, we derive the representations for query (Q), key (K) and value (V) as following:

$$Q, K, V = HW^Q, \ HW^K, \ HW^V, \tag{5}$$

where $W^Q, W^K, W^V \in \mathbb{R}^{C \times C}$ are the learnable linear projections.

Generating dropout target. We aim to generate a vector M' whose elements have probability of being negative infinity, based on the value in M above. Based on the probability values encapsulated within matrix M, we synthesize matrix M'. Each element within $M' \in \mathbb{R}^{n \times C}$ is assigned a value of negative infinity with a probability determined by the corresponding value in M:

$$m'_i = \begin{cases} 0 & with \ probability \ 1 - m_i \\ -inf & with \ probability \ m_i \end{cases} \tag{6}$$

This resultant vector is then expanded across feature channels.

Dropping key process. In preceding step, we derived the dropout rates M'. This is subsequently employed to drop elements from the key vector K. Specifically, by adding the vector M', which contains negative infinity values, to K, the network disregards designated positions, thus achieving a targeted dropout implementation. The mechanism is defined as follows:

$$K' = K + M', \tag{7}$$

The modified token features $T' = \{t'_i | i = 1, ..., n\} \in \mathbb{R}^{n \times C}$ are derived by such formulation:

$$T' = MHA(Q, K', V, PE(\mathcal{D})), \tag{8}$$

where MHA symbolizes the multi-head attention mechanism as delineated in [18], while $PE(\cdot)$ is indicative of the position embedding.

6

Table 1: Classification results on the ModelNet-C dataset, mCE(%, ↓) is reported, the best performance is **bold**. † denotes method designed specifically against corruption.

Method	mCE	Scale	Jitter	Drop-G	Drop-L	Add-G	Add-L	Rotate
DGCNN[TOG2019] [21]	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
PointNet[CVPR2017] [11]	142.2	126.6	64.2	50.0	107.2	298.0	159.3	190.2
PointNet++[NIPS2017] [12]	107.2	87.2	117.7	64.1	180.2	61.4	99.3	140.5
GDANet[AAAI2021] [26]	89.2	83.0	83.9	79.4	89.4	87.1	103.6	98.1
PCT[CVM2021] [4]	92.5	87.2	87.0	52.8	100.0	78.0	138.5	104.2
RPC†[ICML2022] [15]	86.3	84.0	89.2	49.2	79.7	92.9	101.1	107.9
PointNeXt[NIPS2022] [13]	85.6	90.4	129.7	84.7	95.7	25.1	27.6	146.0
PointM2AE[NIPS2022] [31]	83.9	95.7	159.2	53.2	83.6	30.2	30.5	134.9
PointGPT[NIPS2023] [2]	83.4	108.5	102.5	58.5	98.1	29.2	32.7	154.4
APCT (Ours)	**72.2**	94.7	88.3	46.8	85.0	28.5	29.8	132.6
vs. prev. SoTA	↓*11.2*				-			

3.4 Overall Learning Objectives.

We present the adversarial analysis based framework that leverages an Adversarial Significance Identifier along with a Target-guided Promptor. The loss function comprises of the cross-entropy loss $\mathcal{L}_C$ and the auxiliary token identification loss $\mathcal{L}_A$, combined to form the complete loss function:

$$\mathcal{L} = \mathcal{L}_C + \lambda \sum_{s=1}^{S-1} \mathcal{L}_A, \tag{9}$$

where λ is a balancing weight, set as 1. S denotes the total stages in the network.

4 Experiments

We first report our 3D robust classification results on synthetically generated and real-scanned datasets in §4.1 and §4.2, respectively. Subsequently, we assess our methodology efficacy in the robust 3D segmentation in §4.3. In §4.4, we provide ablative analyses on our core algorithm design.

4.1 Results on ModelNet-C

Dataset. We train models on the clean ModelNet40 [22] dataset and evaluate them on the ModelNet-C [15] corruption test suite, which includes seven types of corruptions, "Jitter", "Drop Global/Local", "Add Global/Local", "Scale" and "Rotate", with five levels of severity . We take mean corruption error metric (mCE, %, ↓) as the main evaluation matrix. More details are in the supplementary materials.

Main Results. The comparative performance of various methodologies in terms of their resilience to corruption is systematically encapsulated in Tab. 1. The results unequivocally demonstrate that our proposed method exhibits superior performance, registering an exemplary state-of-the-art mCE score of **72.2%**. This outstanding performance is notably achieved through straightforward network architectural modifications paired with our unique adversarial approach, eschewing the need for intricate training scheme alterations or complex feature extraction. In direct comparison with the sota method PointGPT [2], our method exhibits a marked enhancement of 11.2% mCE reduction. Moreover, APCT consistently attains high mCE scores across categories: 46.8%, 85.0%, 28.5%, and 29.8% for drop-global, drop-local, add-global, and add-local, respectively. These metrics are either at the pinnacle or are approaching the current best results in each respective subcategory. Such results underscore the efficacy of APCT in maintaining robustness against a wide spectrum of corruptions, encompassing both global and local perturbations When compared with RPC [15], an architecture tailored for enhanced robustness against corruption, APCT manifests a striking improvement in mCE for add-global, surpassing 60%. This underlines APCT proficiency in discerning subtle global geometric intricacies within the point cloud, thus facilitating the extraction of pivotal features.

4.2 Results on ScanObjectNN-C

Dataset. ScanObjectNN consists of objects acquired from real-world scans, offering a more authentic evaluation of recongnition, particularly when compared with CAD datasets. ScanObjectNN-C is the corresponding corruption test suite of the hardest version of ScanObjectNN.

Main Results. Tab. 2 summarizes the comparison results on ScanObjectNN-C, showing our algorithm works well on real-world challenging corrupted point clouds. In particular, our algorithm achieves impressive sota result of **74.2%** mCE. In line with the other results, our method distinctly surpasses

7

Table 2: Classification results on the ScanObjectNN-C dataset, mCE(%, ↓) is reported. † denotes method designed specifically against corruption.

Method	mCE	Scale	Jitter	Drop-G	Drop-L	Add-G	Add-L	Rotate
DGCNN[TOG2019] [21]	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
PointNet[CVPR2017] [11]	132.8	141.2	88.4	78.3	125.3	139.2	200.0	156.9
PointNet++[NIPS2017] [12]	96.9	89.7	110.3	55.0	127.7	94.7	90.5	110.7
RPC†[ICML2022] [15]	132.6	131.7	107.3	145.5	130.5	114.2	158.7	140.2
PointNeXt[NIPS2022] [13]	92.1	80.3	107.9	80.7	94.2	94.4	87.5	99.5
PointM2AE[NIPS2022] [31]	78.2	68.5	119.3	55.3	83.2	35.7	71.4	114.2
PointGPT[NIPS2023] [2]	84.7	119.9	104.0	58.5	76.9	36.1	72.2	125.1
APCT (Ours)	**74.2**	104.5	98.9	48.9	67.0	30.0	63.0	107.1
vs. prev. SoTA	*↓4.0*				-			

them, delivering **4.0%** and **7.5%** mCE reduction for PointM2AE [31] and PointGPT [2], respectively. This confirms our algorithm is applicable in real-world corruptions. Notably, our method obtains consistent high performance of nearly all categories. it achieves 48.9%, 67.0%, 30.0%, and 63.0% mCE over drop-global, drop-local, add-global, and add-local, respectively. These results are all the sota results in each subcategory, illustrating that our method performs well in the face of point adding and removing corruption, both globally and locally.

4.3 Results on Shapenet-C

Dataset. Our method can generalize well to down-stream tasks, including part segmentation. To validate this, we conduct a part segmentation experiment on the ShapeNet-C dataset [14].

Main Results. Tab. 3 showcases the performance of our method, achieving a mCE score of **81.4%** and surpassing the previous sota GDANet and PointMAE by an impressive margin of 10.9% and 11.3%, respectively. It is worth highlighting that PointMAE operates as a scheme of masked autoencoders for point cloud self-supervised learning, which is pretrained on large set of data. This underscores the substantial enhancement brought about by our technique in improving the model robustness

Table 3: Segmentation results in terms of mCE (%, ↓) on ShapeNet-C dataset.

Method	mCE	Scale	Jitter	Drop-G	Drop-L	Add-G	Add-L	Rotate
DGCNN[TOG2019] [21]	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
PointNet[CVPR2017] [11]	117.8	108.2	105.0	98.3	113.2	138.6	117.3	143.8
PointNet++[NIPS2017] [12]	111.2	95.0	108.1	85.6	198.3	88.6	108.3	94.7
PAConv[CVPR2021] [25]	92.7	92.7	107.2	92.5	92.7	74.3	94.8	94.8
GDANet[AAAI2021] [26]	92.3	92.2	101.2	94.2	94.6	71.2	95.7	96.9
PT[ICCV2021] [32]	104.9	107.6	107.2	103.2	108.1	111.2	106.6	90.7
PointMLP[ICLR2022] [9]	97.7	96.5	113.2	88.7	99.1	92.9	106.1	**87.6**
Point-BERT[CVPR2022] [29]	103.3	93.8	109.8	87.3	92.7	117.0	119.9	102.5
Point-MAE[ECCV2022] [10]	92.7	**90.8**	103.5	**85.2**	**88.2**	77.6	103.1	100.3
APCT (Ours)	**81.4**	93.9	109.9	86.6	93.4	**40.1**	**41.6**	103.8
vs. prev. SoTA	*↓10.9*				-			

4.4 Ablation Study

To validate the efficacy of our core algorithm designs and parameter settings, we conduct a series of ablative studies on ModelNet-C [15].

Effect of adversarial dropping strategy. To ascertain the impact of our core idea of adversarial dropping, we conducted an ablation study by removing the adversarial dropping process, along with the collaborative supervising identification process. As shown in Tab. 4, the baseline model, trained in the standard strategy, gains 76.2% and 78.8% mCE, on ModelNet-C and ScanObjectNN-C, respectively. Additionally considering the dropping strategy and supervising identification process $\mathcal{L}_A$(Eq. 3) can lead to 4.0% and 4.6% mCE reduction, respectively. However, adding dropping strategy without ancillary constraints from $\mathcal{L}_A$ will lead to mCE increasing. These results verify that combining these two training objectives can yield the best results, indicating that mining data structures from subsidiary component can benefit detailed analysis of point cloud

Visualization of Data Distribution. To showcase the efficacy of our algorithm in mitigating model overfitting to particular patterns, we conduct an analysis on the statistical distribution of the patterns learned by the model. Specifically, we compute the variance of patterns learned from Level 2 Jitter

8

Table 4: Ablation of Main Components in our method.

Drop	$\mathcal{L}_A$	Robust Cls.	
		MN-C	SCNN-C
✗	✗	76.2	78.8
✗	✓	74.3	76.5
✓	✗	77.8	80.5
✓	✓	**72.2**	**74.2**

Table 5: Ablation of mapping ratio γ.

γ	mCE
[0.1, 0.1, 0.1]	78.1
[0.2, 0.2, 0.2]	**72.2**
[0.3, 0.3, 0.3]	76.5
[0.1, 0.2, 0.3]	77.4
[0.3, 0.2, 0.1]	76.9

Figure 6: Ablation of selection number k

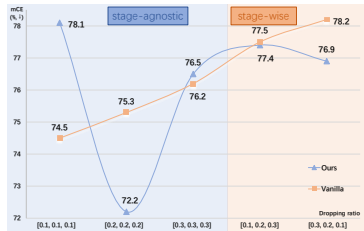



Figure 7: **Comparative Curves** of our algorithm and vanilla dropout strategy.

Figure 8: Visualization of patterns learned by the classifier on ModelNet-C.

Figure 9: Statistical variance distribution of patterns learned.

within the ModelNet-C dataset, as depicted in Fig. 9. Each point in the figure corresponds to the variance from an individual sample from the dataset, the radius represents the value. Analysis reveals that the pattern distribution associated with the vanilla dropout approach tends to exhibit increased variance, which suggests a propensity for the model to overfit to certain patterns. In contrast, APCT prompts the model to integrate information from a broader array of tokens.

Effect of selection number k. We next investigate the impact of the selection number k of Adversarial Significance Identifier in Fig. 6. Here $k = 1$ means directly treating the max-feat token as the single essential part. This baseline ($k = 1$) obtains 75.1% mCE on ModelNet-C. After more essential pattern mining, we observe improvement against corruption, *e.g.*, 75.1% → 72.2% mCE when $k = 2$. When $k > 2$, further increasing k gives marginal performance gains even worse results. We speculate this is because the model is distracted by some trivial patterns due to over-mining.

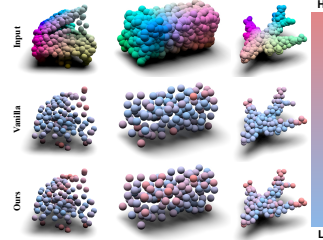
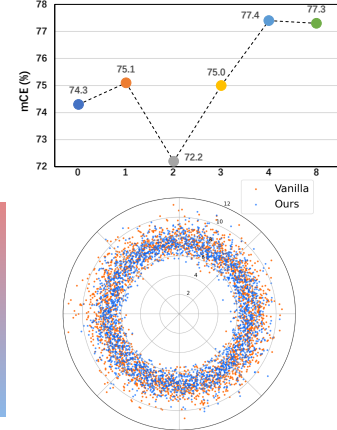
Effect of mapping rate γ. Tab. 5 gives the performance with regard to the ratio γ in mapping function (in Eq. 4). The model performs better with a medium ratio $[0.2, 0.2, 0.2]$, showing that moderate mapping ratio is more favored. Moreover, at the asymptotic cases of $[\gamma_1, \gamma_2, \gamma_3] = [0.1, 0.2, 0.3]$ or $[\gamma_1, \gamma_2, \gamma_3] = [0.3, 0.2, 0.1]$, the performance drops considerably, evidencing that gradually adjusting the dropping strategy is not a sound solution.

Disparity between ours and vanilla dropout. Our method diverges distinctly from the vanilla dropout, from the perspectives of motivation, technical implementation, and empirical outcomes. ❶ From Motivation: we aim to address a nuanced challenge overlooked by standard dropout method: the tendency of deep learning models to make misguided inferences from local features within perturbed samples, particularly prevalent in real-world data scenarios. ❷ From Implementation: dropout fosters generalization by randomly attenuating network parameters. In contrast, our approach eschews simplistic random dropout in favor of a meticulously engineered mechanism that identifies and diminishes tokens deemed pivotal by the model in its preliminary assessments, thereby compelling the network to recalibrate and shift its focus to previously overlooked tokens, as seen in Fig. 2a and Fig. 12. ❸ For results, curves in Fig. 7 demonstrates that when contending with corruptions in the ModelNet-C dataset, our method significantly outperforms the standard dropout (**72.2%** vs 74.5%).

5 Conclusion

In this work, we introduce a new algorithm, tailored for point cloud recognition in the presence of real-world corruptions. Our algorithm incorporates an adversarial dropping strategy, facilitating the capture of diverse patterns, enabling the assimilate information from non-corrupted regions, thereby ensuring robust prediction under local pattern damaged scenario. Experimental evaluations on comprehensive benchmarks manifest its superiority.

Limitations. While current method can discern different patterns through adversarial strategies and render accurate judgments under corrupted scenarios, the optimal utilization of these cues has not been extensively explored in this paper. In future work, we intend to delve deeper into this aspect.

9

Acknowledgments This work was financially supported by the National Natural Science Foundation of China (No. 62101032), the Postdoctoral Science Foundation of China (Nos. 2021M690015, 2022T150050), and Beijing Institute of Technology Research Fund Program for Young Scholars (No. 3040011182111).

References

[1] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:1512.03012, 2015.

[2] Guangyan Chen, Meiling Wang, Yi Yang, Kai Yu, Li Yuan, and Yufeng Yue. Pointgpt: Auto-regressively generative pre-training from point clouds. Advances in Neural Information Processing Systems, 36, 2024.

[3] Yunlu Chen, Vincent Tao Hu, Efstratios Gavves, Thomas Mensink, Pascal Mettes, Pengwan Yang, and Cees GM Snoek. Pointmixup: Augmentation for point clouds. In ECCV, pages 330–345. Springer, 2020.

[4] Meng-Hao Guo, Jun-Xiong Cai, Zheng-Ning Liu, Tai-Jiang Mu, Ralph R Martin, and Shi-Min Hu. Pct: Point cloud transformer. Computational Visual Media, 7(2):187–199, 2021.

[5] Sihyeon Kim, Sanghyeok Lee, Dasol Hwang, Jaewon Lee, Seong Jae Hwang, and Hyunwoo J Kim. Point cloud augmentation with weighted local transformations. In ICCV, pages 548–557, 2021.

[6] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

[7] Dogyoon Lee, Jaeha Lee, Junhyeop Lee, Hyeongmin Lee, Minhyeok Lee, Sungmin Woo, and Sangyoun Lee. Regularization strategy for point cloud via rigidly mixed sample. In CVPR, pages 15900–15909, 2021.

[8] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. arXiv preprint arXiv:1608.03983, 2016.

[9] Xu Ma, Can Qin, Haoxuan You, Haoxi Ran, and Yun Fu. Rethinking network design and local geometry in point cloud: A simple residual mlp framework. In ICLR, 2022.

[10] Yatian Pang, Wenxiao Wang, Francis EH Tay, Wei Liu, Yonghong Tian, and Li Yuan. Masked autoencoders for point cloud self-supervised learning. In ECCV, pages 604–621. Springer, 2022.

[11] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In CVPR, 2017.

[12] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In NIPS, 2017.

[13] Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Abed Al Kader Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. In NIPS, 2022.

[14] Jiawei Ren, Lingdong Kong, Liang Pan, and Ziwei Liu. Pointcloud-c: Benchmarking and analyzing point cloud perception robustness under corruptions. preprint, 2022.

[15] Jiawei Ren, Liang Pan, and Ziwei Liu. Benchmarking and analyzing point cloud classification under corruptions. arXiv preprint arXiv:2202.03377, 2022.

[16] Tzungyu Tsai, Kaichen Yang, Tsung-Yi Ho, and Yier Jin. Robust adversarial objects against deep learning models. In AAAI, volume 34, pages 954–962, 2020.

10

[17] Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Thanh Nguyen, and Sai-Kit Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In ICCV, 2019.

[18] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. NIPS, 30, 2017.

[19] Jie Wang, Lihe Ding, Tingfa Xu, Shaocong Dong, Xinli Xu, Long Bai, and Jianan Li. Sample-adaptive augmentation for point cloud recognition against real-world corruptions. In ICCV, 2023.

[20] Jie Wang, Jianan Li, Lihe Ding, Ying Wang, and Tingfa Xu. Papooling: Graph-based position adaptive aggregation of local geometry in point clouds. arXiv preprint arXiv:2111.14067, 2021.

[21] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. TOG, 38(5):1–12, 2019.

[22] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In CVPR, 2015.

[23] Ziyi Wu, Yueqi Duan, He Wang, Qingnan Fan, and Leonidas J Guibas. If-defense: 3d adversarial point cloud defense via implicit function based restoration. arXiv preprint arXiv:2010.05272, 2020.

[24] Chong Xiang, Charles R Qi, and Bo Li. Generating 3d adversarial point clouds. In CVPR, pages 9136–9144, 2019.

[25] Mutian Xu, Runyu Ding, Hengshuang Zhao, and Xiaojuan Qi. Paconv: Position adaptive convolution with dynamic kernel assembling on point clouds. In CVPR, 2021.

[26] Mutian Xu, Junhao Zhang, Zhipeng Zhou, Mingye Xu, Xiaojuan Qi, and Yu Qiao. Learning geometry-disentangled representation for complementary understanding of 3d object point cloud. In AAAI, pages 3056–3064, 2021.

[27] Xu Yan, Chaoda Zheng, Zhen Li, Sheng Wang, and Shuguang Cui. Pointasnl: Robust point clouds processing using nonlocal neural networks with adaptive sampling. In CVPR, 2020.

[28] Xianggang Yu, Mutian Xu, Yidan Zhang, Haolin Liu, Chongjie Ye, Yushuang Wu, Zizheng Yan, Chenming Zhu, Zhangyang Xiong, Tianyou Liang, et al. Mvimgnet: A large-scale dataset of multi-view images. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 9150–9161, 2023.

[29] Xumin Yu, Lulu Tang, Yongming Rao, Tiejun Huang, Jie Zhou, and Jiwen Lu. Point-bert: Pre-training 3d point cloud transformers with masked point modeling. In CVPR, pages 19313–19322, 2022.

[30] Jinlai Zhang, Lyujie Chen, Bo Ouyang, Binbin Liu, Jihong Zhu, Yujin Chen, Yanmei Meng, and Danfeng Wu. Pointcutmix: Regularization strategy for point cloud classification. Neurocomputing, 505:58–67, 2022.

[31] Renrui Zhang, Ziyu Guo, Peng Gao, Rongyao Fang, Bin Zhao, Dong Wang, Yu Qiao, and Hongsheng Li. Point-m2ae: multi-scale masked autoencoders for hierarchical point cloud pre-training. Advances in neural information processing systems, 35:27061–27074, 2022.

[32] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In ICCV, 2021.

[33] Tianhang Zheng, Changyou Chen, Junsong Yuan, Bo Li, and Kui Ren. Pointcloud saliency maps. In ICCV, pages 1598–1606, 2019.

[34] Hang Zhou, Kejiang Chen, Weiming Zhang, Han Fang, Wenbo Zhou, and Nenghai Yu. Dup-net: Denoiser and upsampler network for 3d adversarial point clouds defense. In CVPR, pages 1961–1970, 2019.

11

A   Appendix / supplemental material

The supplementary material herein extends the discussion and analysis presented in the primary manuscript. It is structured as follows:

Training Setup Details. (§ A.1) : This section delves into the details of our training methodology, offering a comprehensive understanding of our training process.

Computational Overhead. (§ A.2): This section introduces the impact of our method on computational overhead.

Results on Clean dataset. (§ A.3) This section introduces the performance of our method on uncorrupted point cloud dataset.

Results on Point Cloud Attack Defense. (§ A.4) This section introduces the ability of our method on point cloud attack defense.

Results on MVImageNet. (§ A.5) This section introduces the performance of our method on more challenging real-world scanned dataset MVImageNet.

Performance of APCT with augmentation methods. (§ A.6) This section introduces the performance of our method with various augmentation methods.

Effect of the adversarial mechanism with different tokenization methods. (§ A.7) This section introduce the effect of our adversarial mechanism with different tokenization methods.

Effect of the adversarial mechanism incorporating to advanced methods. (§ A.8) This section introduce the effect of our adversarial mechanism incorporating to advanced methods.

Corruption Implementation Details. (§ A.9) : Detailed explication of the corruption settings employed within benchmark datasets is provided, elucidating the experimental conditions and variables.

Focal Tokens Identification Implementation. (§ A.10) : The algorithm for identifying focal tokens is articulated, including a pseudo-code representation in a Pytorch-like syntax, to facilitate replicability and clarity in implementation.

Comparative Analysis of Confusion Matrices. (§ A.11) : This section presents a comparative study of the confusion matrices resulting from standard transformer-based point cloud models vis-à-vis our proposed algorithm, thereby highlighting the distinctive attributes and performance metrics.

More Visualizations of Learned Patterns. (§ A.12) : Further visual evidence is furnished, showcasing the patterns discerned by the classifier's final self-attention layer, thus reinforcing the interpretability aspect of our model.

Extended Results. (§ A.13) : Comprehensive results pertaining to ModelNet-C, ScanObjectNN-C, and ShapeNet-C are presented, augmenting the main findings with additional experimental results.

A.1   Training Setup Details

ModelNet-C. Our model is optimized using the AdamW optimizer [6] with a batch size of 32 for 300 epochs. The optimizer utilizes a learning rate of 0.0005 and weight decay of 0.05. The CosineAnnealingLR [8] scheduler is employed to decrease the learning rate to the minimum value of 1e-6, and the warm up epochs is set to 10. On ModelNet, input point cloud is partitioned into 64 tokens with 384 dimension of feature.

ScanObjectNN-C. On ScanObjectNN, our model is trained with a batch size of 32 using the AdamW optimizer for 300 epochs. The optimizer is configured with a learning rate of 0.0005 and weight decay of 0.05. The learning rate is decayed to the minimum value of 1e-6 using the CosineLRScheduler scheme, with warp up epochs of 10. For the representation, the point cloud data is systematically partitioned into 128 tokens, each having a feature dimension of 384.

ShapeNet-C. In this experiment, the proposed model is trained for 300 epochs using the AdamW optimizer. A batch size of 16 is set for the training process. The optimizer was configured with a learning rate of 0.0002 and weight decay of 0.05. Additionally, the learning rate was decayed to the minimum value of 1e-6 by employing the CosineLRScheduler scheme.

12

Evaluation metrics. We used evaluation metrics (mCE, mOA and RmCE) as specified in previous work RPC [15].

A.2 Computational Overhead

To demonstrate the computational impact of our approach, we also conducted corresponding comparisons, experiments are conducted on one GeForce RTX 3090. Its impact on computational overhead is presented in Tab. 6, including deliberations on both training and inference speed. Noteworthy is that our method stands out for its exemplary memory efficiency, as it refrains from incurring any additional memory overhead. When juxtaposed with the baseline technique, our method, albeit registering a marginal deceleration in the realms of training (~ 32 samples/s, 6% delay) and inference (~ 66 samples/s, 6% delay), emerges superior with a pronounced decrement in mCE, registering a downturn of 4.0%.

Table 6: Analysis of computational overhead.

Method	Memory (G)	Train speed (samples/s)	Infer speed (samples/s)	mCE (%)
Baseline	10.9	415.2	1111.7	76.2
+Ours	10.9	383.4	1045.8	**72.2**
$\triangle$	-	$\downarrow\sim6\%$	$\downarrow\sim6\%$	$\downarrow4.0$

A.3 Results on Clean Dataset

In pursuit to rigorously evaluate and validate the potency of our proposed technique, we also conduct comprehensive evaluations on a pristine dataset. Specifically, we chose the ScanObjectNN's [17] most challenging variant, PB-T50-RS, for our experiments. Our proposed technique consistently exhibited improvements even on such clean datasets, a testament underscored by our exhaustive experimental validation. The results presented in Tab. 7 show a discernible performance boost, moving from 85.3% to 86.2%. Notably, this is a highly competitive result compared with many advanced methods [13, 9], demonstrating the generalizability of our proposed algorithm.

Table 7: Results on ScanObjectNN, OA($\%$, $\uparrow$) is reported. Note: No multi-scale inference is employed. Baseline denotes model w/o Adversarial Dropping.

Method	OA
DGCNN [21]	78.1
PointNet [11]	68.2
PointNet++ [12]	77.9
PointMLP [9]	85.4
RPC [15]	74.7
PointNeXt [13]	**87.3**
Baseline	85.3
+ Ours	86.2 $\uparrow0.9$

A.4 Results on Point Cloud Attack Defense

Setup. Point cloud attack for classification aims to manipulate the input point cloud in a manner that induces misclassification by a well-tuned classifier. Encouraged by the remarkable performance of our algorithm against corruption, we investigate its potential application in point cloud defense. To evaluate this, we assess the performance of baseline model trained using our method on various attacks, including point perturbation attack [24], individual point adding attack [24], kNN attack [16], and point dropping attack [33]. Consistent with previous works [23, 34], we conduct targeted attacks and report the resulting classification accuracy. Higher accuracy indicates better defense against attack.

Main Results. Based on the results presented in Tab. 8, it is observed that our algorithm consistently perform well on defending all attacks, especially on two point dropping attacks. The accuracy im-

13

Table 8: Results on point cloud attack defense, OA(%, ↑).

Method	Perturb	Add-CD	Add-HD	kNN	Drop-100	Drop-200
No Defense	0.00	0.00	0.12	0.00	80.55	69.53
APCT(Ours)	74.31	72.85	54.82	49.96	84.08	76.82
△	*+74.31*	*+72.85*	*+54.70*	*+49.96*	*+3.53*	*+7.29*

provement can be as high as 7.29% in the Drop-200 attack, indicating the scalability and effectiveness of the approach. For point cloud attack defend, our method perform well on defending all attacks, as seen in Tab. 8. Even with the Perturb attack, which introduces very fine perturbations leading to a chaotic point cloud, APCT achieves a good performance of 74.31% OA. These results evident the strong generalization ability of our APCT. The above analysis leads to the conclusion that our method has a strong generalization ability across various point cloud attack algorithms.

A.5 Results on MVImageNet

We further evaluate the performance of our approach on one additional dataset MVImageNet [28]. It is a challenging benchmark for real-world point cloud classification, which contains 64,000 training and 16,000 testing samples. As shown in the Tab. 9, our approach achieves 86.6% OA and still exhibits good generalization capacity in real-world scenarios.

Table 9: Classification results on the MVIamgeNet dataset, OA(%, ↑) is reported.

Method	OA (%, ↑)
PointNet [11]	70.7
PointNet++ [12]	79.2
DGCNN [21]	86.5
PAConv [25]	83.4
PointMLP [9]	88.9
APCT (Ours)	86.6

A.6 Performance of APCT with augmentation methods

We further evaluate the performance of our approach with various data augmentation methods on ModelNet-C dataset, the results are as follows. Beside the discussed PointMixup [3] and PointCut-Mix [30], we additionally incorporate experiments with the data augmentation techniques Point-WOLF [5], RSMix [7], and WOLFMix [15].

Among these, PointMixup, PointCutMix and RSMix fall under the category of mixing augmentation, where they mix several point clouds following pre-defined regulations. PointWOLF pertains to deformation techniques that non-rigidly deforms local parts of an object. WOLFMix combines both mixing and deformation augmentations, which first deforms the object, and subsequently rigidly mixes the deformed objects together.

As shown in the Tab. 10, data augmentation methods further improve the robustness of our method against point cloud corruptions. Employing mixing or deformation data augmentation techniques independently can enhance the robustness of the model, *e.g.*, the results of our model with PointWOLF (67.0% mCE) and with PointMixup (66.2% mCE). When these two techniques are combined, as in WOLFMix, the robustness of the model is further augmented (64.7% mCE). Additionally, these experiments demonstrate the compatibility of our method with various data augmentation techniques, further underscoring its potential in addressing data corruption.

A.7 Effect of the adversarial mechanism with different tokenization methods

To evaluate the generalization and robustness of our method, we apply our method with two alternative tokenization methods: mini-DGCNN and mini-PCT. The results are shown in the Tab. 11.

14

Table 10: Performance of APCT with augmentation methods on ScanObjectNN-C.

Method	mCE (%, ↓)
APCT (Ours)	72.2
+ PointMixup [3]	66.2
+ PointCutMix-R [30]	69.7
+ PointWOLF [5]	67.0
+ RSMix [7]	71.3
+ WOLFMix [15]	64.7

Table 11: Results of the adversarial mechanism with different tokenization methods.

Method	mCE (%, ↓)
mini-DGCNN	71.1
mini-PCT	72.4
mini-PointNet (Ours)	72.2

As we can see from the results, the performance of our adversarial mechanism is relatively stable across different tokenization methods. This indicates that our method is not sensitive to the specific tokenization method used and can effectively improve the robustness of point cloud models.

A.8 Effect of the adversarial mechanism incorporating to advanced methods

We have extended the adversarial 'digging sub-optimal patterns' mechanism to two state-of-the-art (SOTA) methods, PointM2AE [31] and PointGPT [2], on the ModelNet-C dataset. The results are promising and demonstrate the general applicability of our approach.

As shown in the Tab. 12 below, incorporating our 'digging sub-optimal patterns' mechanism into PointM2AE and PointGPT resulted in significant reduction in mCE scores. These results suggest that our approach can effectively enhance the robustness of various point cloud recognition models. By encouraging the model to explore and utilize a broader range of patterns, our method enables the models to better generalize to corrupted data.

Table 12: Results of the adversarial mechanism incorporating to advanced methods.

Method	mCE (%, ↓)
PointM2AE	83.9
$+Ours$	82.9 ↓1.0%
PointGPT	83.4
$+Ours$	82.0 ↓1.4%

A.9 Corruption Implementation Details

In the realm of corruption benchmarks, ModelNet-C [15], ScanObjectNN-C [19], and ShapeNet-C [14] are prominent datasets, as detailed in Tab. 13. Among these, ScanObjectNN-C is particularly noteworthy for its real-world scanned dataset, presenting a heightened level of challenge. Central to the analysis of these benchmarks is the application of seven atomic corruptions, which serve as fundamental perturbations for evaluating algorithmic resilience in the face of data degradation. Details about the corruptions are as follows:

- *Scale*: Application of random anisotropic scaling to the point cloud.
- *Rotate*: Rotation of the point cloud by a small angle.
- *Jitter*: Addition of Gaussian noise to point coordinates.
- *Drop-Global*: Random removal of points from the point cloud.
- *Drop-Local*: Random elimination of several local clusters from the point cloud.
- *Add-Global*: Addition of random points sampled within a unit sphere.
- *Add-Local*: Expansion of random points on the point cloud into normally distributed clusters.

15

Table 13: Point cloud corruption benchmarks comparison.

Dataset	Refernce	Number Samples	Real-world
ModelNet-C [15]	ICML2022	2468	✗
ShapeNet-C [14]	Arxiv2022	2874	✗
ScanObjectNN-C [19]	ICCV2023	2882	✓

Clean   Scale   Jitter   Drop-Global   Drop-Local   Add-Global   Add-Local   Rotate

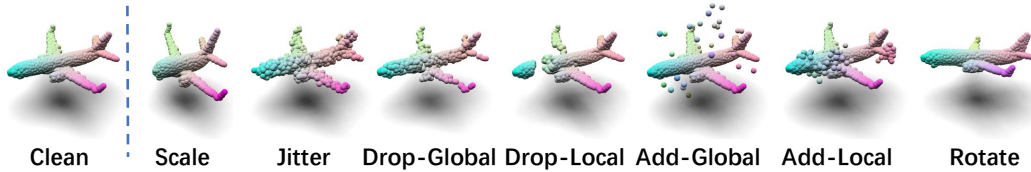



Figure 10: Visualization of samples in ModelNet-C, which is constructed by seven types of corruptions with five levels of severity. Listed examples are from severity level 2.

Each corruption type is associated with five severity levels, facilitating a comprehensive evaluation of robustness. This diverse set of corruptions serves to systematically assess the resilience of models against various perturbations in the point cloud data. The differential attributes between samples subjected to an array of corruptions and the clean sample are visually delineated in Fig. 10. An observable shift in structural fidelity is evident in the corrupted instances when juxtaposed against their unaltered originals. These perturbations markedly hinder the performance efficacy of computational models, culminating in a consistent decline in model accuracy across a spectrum of corruption categories.

A.10 Focal Tokens Identification Implementation

Algorithm. 1 shows the implementation of focal tokens identification introduced in the main manuscript. Through such implementation, we can easily calculate all token significances associated to overall perception and identify the focal ones.

Algorithm 1 Pseudo-Code of *Identifying focal tokens* in a Pytorch-like Style.

```
Input:
  tokens - [N, C], where N denotes the length and C denotes the feat
      dimensions.
  k - number of focal tokens selected in each feat channel, default 2.
Output:
  matrix - [N], each value within matrix enumerates number within the
      range (0,C), indicating the token significance associated to overall
      perception.

# identify focal tokens
Function: identify_focaltokens(tokens, k):
   N, C = tokens.shape
   # sort token in each feat channel
   _, sort_idx = sort(tokens, dim=0, descending=True)
   # select the idxs of top k tokens
   idx_topk = sort_idx[:k, :]
   idx_topk = idx_topk.view(-1)
   matrix = zeros(N)
   # accumulate the idxs
   matrix.scatter_add(1, idx_topk, ones_like(idx_topk))
   return matrix
```

16

A.11 Comparative Analysis of Confusion Matrices

To ascertain the efficacy of our method on corrupted data, we selected all corruption types at an intermediate level (level 2) from ModelNet-C [15] for comparative experiments. As illustrated in Fig. 11, we present side-by-side confusion matrix comparisons. Notably, our adversarial dropping strategy substantially enhances the model's performance under corruption. This suggests that our approach incentivizes the model to delve into a broader spectrum of patterns, ultimately converging to a global motif. Consequently, even if specific local motifs deteriorate within corrupted data, the model retains the capability to glean information from alternative regions for proficient predictions.

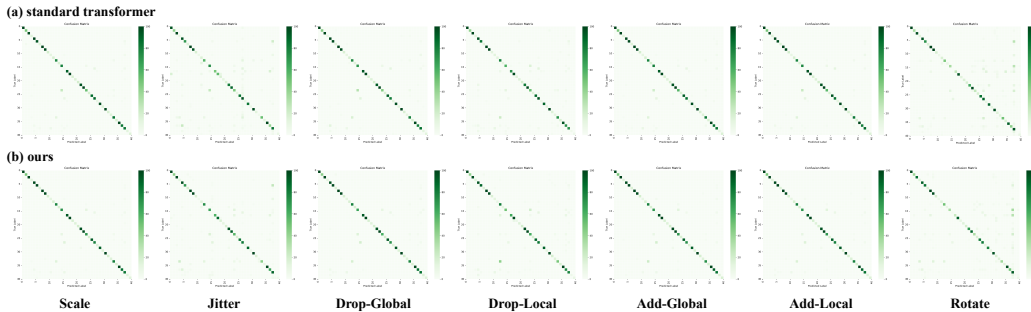



Figure 11: Side-by-side comparison of the confusion matrices for two distinct architectures: the standard transformer-based point cloud model (top row) and ours (bottom row).

A.12 More Visualizations of learned patterns

To elucidate the implications of our adversarial dropping strategy on real-world corruptions, we embarked on a methodical analysis, juxtaposing token features that the classifier attends to, particularly on the perturbed data from the jitter-2 test suite of ModelNet-C. Fig. 12 offers a visual delineation of the classifier's final self-attention layer. Specifically, we adopt the *Focal tokens identification* procedure, as outlined in the main manuscript, post-normalization. The computed token features are then harnessed to epitomize their significance to the model. Tokens rendering substantial contributions to the perception model are depicted with heightened color intensities: red symbolizing high contribution, while blue denotes low contribution. The visualizations indicate that, standard architectures tend to overfit to specific local patterns, such as table legs or airplane wings. In contrast, our approach fosters a more comprehensive exploration of patterns, culminating in the capture of a global motif. Hence, even if some local motifs degrade in corrupted data, the model can still extract valuable information from other regions for effective prediction.

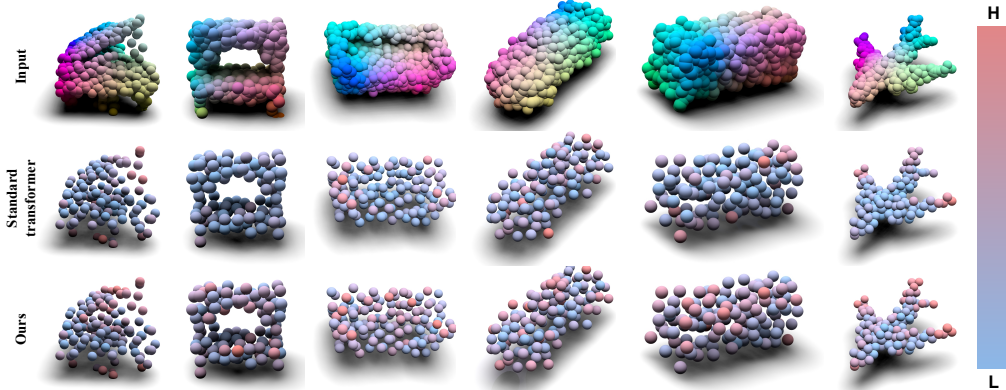



Figure 12: More visualization results of patterns learned by the classifier on ModelNet-C. Tokens with high / low scores are in red / blue, respectively.

17

A.13 Extended Results

ModelNet-C. We present complete results for both the mOA and RmCE metrics in Tab. 14 and Tab. 15, respectively.

ScanObjectNN-C. We show full results for the mOA metric and the RmCE metrics in Tab. 16 and Tab. 17, respectively.

ShapeNet-C. The complete results of the RmCE metric are presented in Tab. 18 for ShapeNet-C dataset.

Table 14: Classification results on the ModelNet-C dataset, mOA($\%, \uparrow$) is reported, the best performance is **bold**. † denotes method designed specifically against corruption.

Method	mOA	Scale	Jitter	Drop-G	Drop-L	Add-G	Add-L	Rotate
DGCNN[TOG2019] [21]	76.4	90.6	68.4	75.2	79.3	70.5	72.5	78.5
PointNet[CVPR2017] [11]	65.8	88.1	79.7	87.6	77.8	12.1	56.2	59.1
PointNet++[NIPS2017] [12]	75.1	91.8	62.8	84.1	62.7	81.9	72.7	69.8
GDANet[AAAI2021] [26]	78.9	92.2	73.5	80.3	81.5	74.3	71.5	78.9
PAConv[CVPR2021] [25]	73.0	91.5	53.7	75.2	79.2	68.0	64.3	79.2
PCT[CVM2021] [4]	78.1	91.8	72.5	86.9	79.3	77.0	61.9	77.6
RPC†[ICML2022] [15]	79.5	92.1	71.8	87.8	83.5	72.6	72.2	76.8
PointNeXt[NIPS2022] [13]	80.5	91.5	59.0	79.0	80.2	92.6	92.4	68.6
PointM2AE[NIPS2022] [31]	80.6	91.0	49.7	86.8	82.7	91.1	91.6	71.0
PointGPT[NIPS2023] [2]	81.7	89.8	67.6	85.5	79.7	91.4	91.0	66.8
APCT (Ours)	**84.1**	91.1	72.1	88.4	82.4	91.6	91.8	71.5
vs. prev. SoTA	↑*2.4*				-			

Table 15: Classification results on the ModelNet-C dataset, RmCE($\%, \downarrow$) is reported, the best performance is **bold**. † denotes method specifical against corruption.

Method	RmCE	Scale	Jitter	Drop-G	Drop-L	Add-G	Add-L	Rotate
DGCNN[TOG2019] [21]	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
PointNet[CVPR2017] [11]	148.8	130.0	45.5	17.8	97.0	355.7	171.6	224.1
PointNet++[NIPS2017] [12]	111.4	60.0	124.8	51.1	227.8	50.2	101.0	164.5
GDANet[AAAI2021] [26]	86.5	60.0	82.2	75.3	89.5	86.4	109.0	102.8
PAConv[CVPR2021] [25]	121.1	105.0	164.9	105.7	108.3	115.8	145.8	102.1
PCT[CVM2021] [4]	88.4	60.0	84.7	35.1	103.0	72.4	154.7	109.2
RPC†[ICML2022] [15]	77.8	45.0	87.6	29.9	71.4	92.3	103.5	114.9
PointNeXt[NIPS2022] [13]	76.6	55.0	138.8	78.2	93.2	0.0	1.0	170.2
PointM2AE[NIPS2022] [31]	57.7	5.0	171.1	24.7	63.2	0.0	-2.5	142.6
PointGPT[NIPS2023] [2]	68.1	80.0	98.3	33.9	88.0	0.0	2.0	174.5
APCT (Ours)	**51.4**	40.0	81.8	20.1	71.4	1.4	0.5	144.7
vs. prev. SoTA	↓*6.3*				-			

18

Table 16: Classification results on the ScanObjectNN-C dataset, mOA(%, ↑) is reported. † denotes method designed specifically against corruption.

Method	mOA	Scale	Jitter	Drop-G	Drop-L	Add-G	Add-L	Rotate
DGCNN[TOG2019] [21]	62.8	57.8	45.6	62.2	69.7	54.0	77.3	73.3
PointNet[CVPR2017] [11]	53.3	40.4	51.9	70.3	62.0	35.9	54.6	58.1
PointNet++[NIPS2017] [12]	64.1	62.1	40.0	79.2	61.3	56.4	79.5	70.5
RPC†[ICML2022] [15]	52.2	44.4	41.6	44.9	60.4	47.4	64.0	62.6
PointNeXt[NIPS2022] [13]	65.5	66.1	41.3	69.5	71.4	56.5	80.1	73.4
PointM2AE[NIPS2022] [31]	71.0	71.1	35.1	79.1	74.8	83.6	83.8	69.5
PointGPT[NIPS2023] [2]	68.7	49.4	43.4	77.9	76.7	83.4	83.6	66.6
APCT (Ours)	**72.4**	55.9	46.2	81.5	79.7	86.2	85.7	71.4
vs. prev. SoTA	*↑1.4*				-			

Table 17: Classification results on the ScanObjectNN-C dataset, RmCE(%, ↓) is reported. † denotes method designed specifically against corruption.

Method	RmCE	Scale	Jitter	Drop-G	Drop-L	Add-G	Add-L	Rotate
DGCNN[TOG2019] [21]	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
PointNet[CVPR2017] [11]	106.1	120.2	55.1	15.8	74.8	119.9	228.9	127.7
PointNet++[NIPS2017] [12]	97.6	85.9	114.8	29.7	154.4	93.5	79.1	125.9
RPC†[ICML2022] [15]	102.1	108.2	82.3	126.0	88.8	85.8	126.6	97.3
PointNeXt[NIPS2022] [13]	93.9	75.8	114.4	75.5	98.6	96.7	84.7	111.2
PointM2AE[NIPS2022] [31]	**49.9**	44.6	120.6	19.1	54.7	0.0	0.0	112.8
PointGPT[NIPS2023] [2]	59.7	121.4	99.5	23.3	41.6	0.0	0.0	134.4
APCT (Ours)	56.0	108.2	99.5	19.9	40.4	0.0	5.9	118.4
vs. prev. SoTA	*↑6.1*				-			

Table 18: Segmentation results in terms of RmCE (%, ↓) on ShapeNet-C dataset.

Method	RmCE	Scale	Jitter	Drop-G	Drop-L	Add-G	Add-L	Rotate
DGCNN[TOG2019] [21]	100.0	100.0	100.0	100.0	100.0	100.0	100.0	100.0
PointNet[CVPR2017] [11]	105.6	35.5	**88.0**	**8.7**	115.2	156.6	120.6	214.4
PointNet++[NIPS2017] [12]	185.0	68.5	132.9	17.6	786.0	83.0	116.9	90.1
PAConv[CVPR2021] [25]	84.8	56.0	133.6	76.4	78.9	59.7	94.7	94.0
GDANet[AAAI2021] [26]	78.5	26.4	111.5	80.6	84.2	53.5	95.2	97.9
PT[ICCV2021] [32]	93.3	98.1	105.1	72.8	107.7	113.3	105.4	**50.7**
PointMLP[ICLR2022] [9]	81.0	47.4	142.8	21.7	96.1	88.2	110.9	60.1
Point-BERT[CVPR2022] [29]	89.5	28.3	135.6	21.3	61.9	130.3	136.0	112.8
Point-MAE[ECCV2022] [10]	70.3	**18.0**	120.9	22.2	**45.9**	65.0	108.8	111.4
APCT (Ours)	**55.3**	39.9	137.3	20.6	70.3	**0.1**	**0.4**	118.5
vs. prev. SoTA	*↓15.0*				-			

19

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims made in the abstract and introduction accurately reflect the contributions and scope.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: See the Section. 5.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

20

Justification: [NA]

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Our implementation details are posted on Section. 3. The data are public datasets and our code will be available.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

21

Answer: [Yes]

Justification: The data are public datasets and our code will be available.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: See Section. 3 and Section. 4 .The data are public datasets and our code will be available.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report details in Section. 4 and do fair comparisons.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

22

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We report these on supplementary materials.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: Research conducted in the paper conform is with the NeurIPS Code of Ethics

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: Our work is only for academic research purpose.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

23

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [No]

Justification: The paper poses no such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: This paper uses existing assets, ModelNet-C [15], ShapeNet-C [14] and ScanObjectNN-C [19]. We cite the creators.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

24

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [No]

Justification: We don't include any new assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [No]

Justification: We didn't use crowdsourcing or conduct research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [No]

Justification: We didn't use crowdsourcing or conduct research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

25