Iterative Transformer Network for 3D Point Cloud

Wentao Yuan David Held Christoph Mertz Martial Hebert
The Robotics Institute
Carnegie Mellon University
{wyuan1, dheld, cmertz, mhebert}@cs.cmu.edu

Abstract

A key challenge in understanding real world 3D data is to learn task-specific features that are invariant or equivariant with respect to geometric transformations. To address this challenge, we propose Iterative Transformer Network (IT-Net), a network module that canonicalizes the pose of a partial object with a series of 3D rigid transformations predicted in an iterative fashion. We demonstrate the efficacy of IT-Net as an anytime pose estimator from partial point clouds without using complete object models. Further, we show that IT-Net achieves superior performance over alternative 3D transformer networks on various tasks, such as partial shape classification and object part segmentation.

1. Introduction

Recently, neural networks operating on point clouds [4, 5] have shown superior performance on 3D understanding tasks such as shape classification and object part segmentation. However, performance on such tasks is often evaluated on complete shapes aligned in a canonical frame [1, 6], while real world 3D data are partial and misaligned. In this work, we tackle the problem of learning from partial, misaligned point clouds using a 3D transformer network.

A key challenge in understanding partial, misaligned point cloud data in the wild is to learn features that are invariant or equivariant with respect to geometric transformations. Spatial Transformer Networks (STN) [3] provide a way to learn such features from images by predicting transformations that align the inputs into a canonical frame which makes subsequent tasks easier. Following the idea of STN, we propose a transformer network that operates on 3D point clouds, named Iterative Transformer Network (IT-Net). IT-Net has two major features that distinguish it from existing 3D transformer networks. First, IT-Net outputs a rigid transformation instead of an affine transformation, which prevents undesirable deformation of the inputs and allows the outputs to be used directly as estimates for object poses. Second, instead of predicting the transformation in a single step, IT-Net takes advantage of an iterative refinement scheme which decomposes a large transformation into smaller ones that are easier to predict. The iterative refinement scheme not only increases accuracy of the predicted transformation, but also allows anytime prediction, where the prediction result can be gradually refined until the test-time computational budget is depleted.

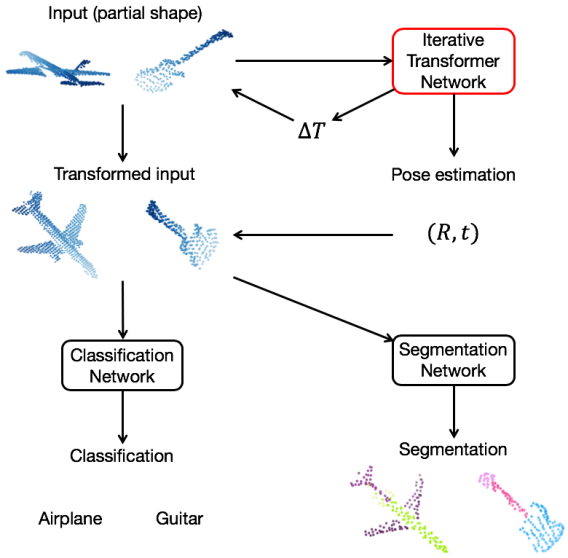

Figure 1: Iterative Transformer Network (IT-Net) predicts rigid transformations from partial point clouds in an iterative fashion. It can be used independently as a pose estimator or jointly with classification and segmentation networks.

We evaluate IT-Net on three point cloud learning tasks – pose estimation, shape classification and part segmentation (Figure 1) – with partial, misaligned inputs from synthetic as well as real world 3D data. We demonstrate that IT-Net can be used as an anytime object pose estimator and show that IT-Net outperforms existing 3D transformer networks when trained jointly with various classification or segmentation networks. We also contribute a dataset consisting of partial point clouds generated from virtual scans of CAD models in ModelNet [6] and ShapeNet [1] as well as real world scans from ScanNet [2]. Our dataset contains challenging inputs with arbitrary 3D rotation, translation and realistic self-occlusion patterns.

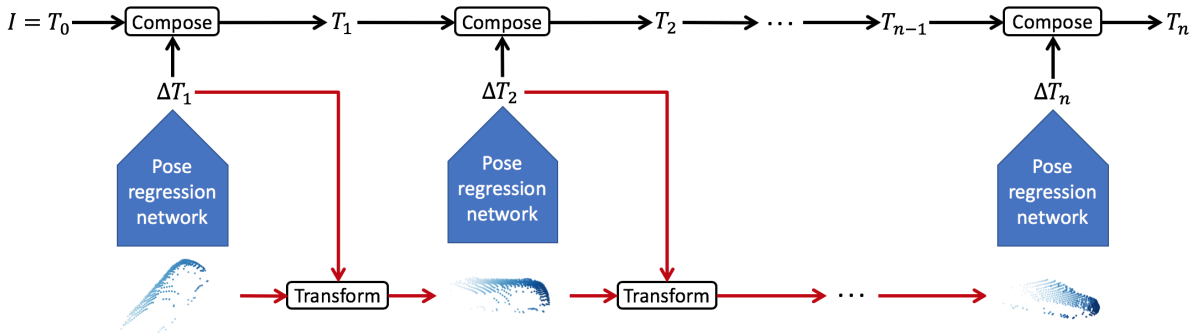

Figure 2: Illustration of the iterative scheme employed by IT-Net. At each iteration, the output of the pose regression network is used to transform the input for the next iteration. The parameters of the pose regression network shown in blue arrows are shared across iterations. The final output is a composition of the transformations predicted at each iteration. Arrows colored in red indicate places where the gradient flow is stopped to decorrelate the inputs at different iterations (see Sec. 2.3).

2. Iterative Transformer Network

Iterative Transformer Network (IT-Net) (Figure 2) takes a 3D point cloud and produces a transformation that can be used directly as a pose estimate or applied to the input before feature extraction for subsequent tasks. In what follows, we introduce two features that differentiate IT-Net from existing 3D transformer networks: the rigid transformation prediction and the iterative refinement scheme.

2.1. Rigid Transformation Prediction

A single iteration of IT-Net is a pose regression network that takes a point cloud and outputs 7 numbers parametrizing a 3D rigid transformation. The first 4 numbers are normalized into a unit quaternion $\mathbf{q}$ and the last 3 are treated as a 3D translation vector $\mathbf{t}$. Then, $\mathbf{q}$ and $\mathbf{t}$ are assembled into a 4×4 matrix $T = \begin{bmatrix} R(\mathbf{q}) & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}$ where $R(\mathbf{q})$ is the rotation matrix corresponding to $\mathbf{q}$. The matrix representation turns the composition of two rigid transformations into a matrix multiplication, which is convenient for composing the outputs from multiple iterations.

In contrast to the affine transformation produced by other 3D transformer networks such as T-Net [4], the rigid transformation predicted by IT-Net can be directly interpreted as a 6D pose, making it possible to use IT-Net independently for pose estimation. More importantly, rigid transformations preserve scales and angles, which is crucial for the iterative application of the network illustrated in Sec. 2.2.

We note that a regularization term $\|AA^T - I\|$ that forces an affine matrix A to be orthogonal can achieve similar effects as predicting a rigid transformation, but our experiments in Sec. 3.2 show that this trick does not give results as good as directly predicting rigid transformations.

2.2. Iterative Refinement

As shown in Figure 2, the forward pass of IT-Net is unfolded into multiple iterations. At the i-th iteration, a rigid transformation ΔT_i is predicted as described in Sec. 2.1. Then, the input is transformed by ΔT_i and the process is repeated. The final output after n iterations is a composition of the transformations predicted at each iteration, which can be written as a simple matrix product $T_n = \prod_{i=1}^{n} \Delta T_i$.

We use a fixed predictor (i.e. share the network's parameters) across different iterations. In addition to reduction in the number of parameters, the fixed predictor allows us to use a different number of unfolded iterations during testing than during training. Consequently, as will be shown in Sec. 3.1, IT-Net can be used as an anytime predictor where increasingly accurate pose estimates can be obtained as the network is applied for more iterations.

The iterative refinement scheme can be interpreted as a way to automatically generate a curriculum, which breaks down the original task into a set of simpler pieces. In earlier iterations, the network learns to predict large transformations that bring the input near its canonical pose. In later iterations, the network learns to predict small transformations that adjusts the estimate from previous iterations. Note that the curriculum is not manually defined but rather generated by the network itself to optimize the end goal.

2.3. Implementation Details

In addition to the key ingredients above, there are a couple of details that are important for the training of IT-Net. First, we initialize the network to predict the identity transformation $\mathbf{q} = [1\ 0\ 0\ 0]$, $\mathbf{t} = [0\ 0\ 0]$. In this way, the default behavior of each iteration is to preserve the transformation predicted by previous iterations. This identity initialization helps prevent the network from producing large transformations which cancel each other. Second, we stop the gradients propagating through input transformations (red arrows in Figure 2) and let the gradients propagate through the output composition only (black arrows in Figure 2). This removes dependency among inputs at different iterations which leads to gradient explosion. Empirical evaluations for these design choices can be found in Sec. 3.1.

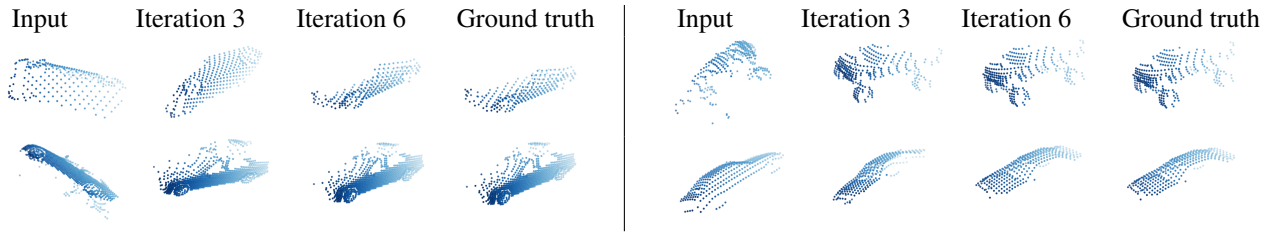

Figure 3: Qualitative results of object pose estimation. The transformed inputs at various iterations are shown above.

3. Experiments

We evaluate IT-Net on various point cloud learning tasks including object pose estimation (Sec. 3.1), shape classification (Sec. 3.2) and object part segmentation (Sec. 3.3). Our experiments are performed on a dataset of object point clouds which captures the incomplete and misaligned nature of real world 3D data. The dataset consists of four parts: ShapeNet Pose, ShapeNet Part, Partial ModelNet40 and ScanNet Objects. The first three parts are generated by rendering depth scans of CAD models in ShapeNet [1] and ModelNet40 [6] from uniformly random viewpoints and back-projecting the scans into point clouds in the camera's coordinates. The last part is obtained by cropping indoor RGBD scans in ScanNet [2] with labeled bounding boxes, where realistic sensor noise and clutter are kept.

3.1. Object Pose Estimation

We investigate the efficacy of the iterative refinement scheme on the task of estimating the canonical pose of an object from a partial observation. Specifically, we use IT-Net to predict the transformation that aligns the input shape to a canonical frame defined across all models in the same category. Unlike most existing works on pose estimation, we do not assume knowledge of the complete object model and we train a single network that generalizes to different objects in the same category.

The network architecture is described in Sec. 2 where the pose regression network is a PointNet [4]. An explicit loss is applied to the output transformation, written as

$$L((R,\mathbf{t}),(\widetilde{R},\tilde{\mathbf{t}})) = \frac{1}{|X|}\sum_{\mathbf{x}\in X}\|(R\mathbf{x}+\mathbf{t})-(\widetilde{R}\mathbf{x}+\tilde{\mathbf{t}})\|_2^2, \quad (1)$$

where $R,\mathbf{t}$ are the ground truth pose and $\widetilde{R},\tilde{\mathbf{t}}$ are the estimated pose and X is the set of input points. We trained IT-Nets on 10,000 car point clouds in ShapeNet Pose and evaluated their performance on 1,000 car point clouds from different car models. Figure 3 shows some qualitative results. In what follows, we provide more detailed analysis.

Number of unfolded iterations The number of unfolded iterations during training is a hyperparameter that controls which iteration to apply the loss. We observed that with too few iterations, the network fails to predict accurate refinements near the canonical pose; with too many iterations, the network becomes overly conservative in its predictions. As shown by Table 1, 5 iterations strike a good balance.

Unfolded iterations	1	2	3	4	5	6	7
no init	17.1	0.5	18.4	7.3	6.2	41.1	13.1
no stop	36.0	5.0	0.0	0.0	4.1	4.1	0.0
ours	36.9	41.7	47.7	61.1	67.6	62.6	48.2

Table 1: Pose accuracy (%) with error threshold $10^\circ, 0.1$ of IT-Nets with different number of unfolded iterations during training. No init means the output is *not* initialized as identity. No stop means the gradient is *not* stopped during input transformations.

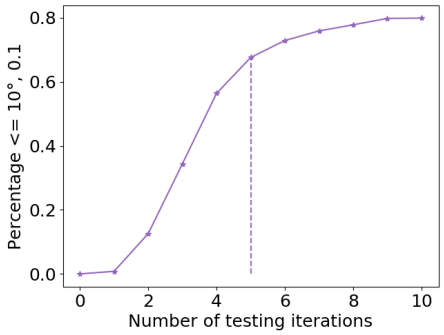

Figure 4: Pose accuracy (%) against the number of iterations applied during inference. The dotted line corresponds to the number of unfolded iterations during training. Note how pose accuracy keeps improving with more testing iterations.

Anytime pose estimation As noted in Sec. 2.2, sharing weights across iterations allows us to use IT-Net as an anytime pose estimator. Figure 4 shows that pose accuracy keeps increasing even when more iterations are applied than the network is trained for. As a result, during inference, we can keep applying the trained IT-Net to obtain increasingly accurate pose estimates until the time budget runs out.

Ablation studies We conducted ablation studies to validate our design choices described in Sec. 2.3. The results are summarized in Table 1. It can be seen that the performance degrades significantly without either identity initialization or gradient stopping.

	mean	table	chair	air plane	lamp	car	guitar	laptop	knife	pistol	motor cycle	mug	skate board	bag	ear phone	rocket	cap
# shapes		5271	3758	2690	1547	898	787	451	392	283	202	184	152	76	68	66	55
# parts		3	4	4	4	4	3	2	2	3	6	2	3	2	3	3	2
None	76.9	78.8	82.6	77.3	71.3	52.3	90.1	76.8	80.0	70.1	40.4	**86.1**	67.6	71.0	66.7	53.1	76.9
T-Net	77.1	79.2	82.5	78.0	70.1	55.7	89.1	73.1	81.5	73.0	39.1	81.1	69.1	**74.1**	71.1	51.4	74.6
IT-Net-1	78.2	79.9	84.3	78.2	**72.9**	54.9	91.0	**78.7**	78.1	71.8	**44.6**	84.8	66.6	71.2	**72.7**	**55.0**	77.9
IT-Net-2	**79.1**	**80.2**	**84.7**	**79.9**	72.1	**62.6**	**91.1**	76.4	**82.8**	**76.9**	44.0	84.4	**71.8**	68.1	66.8	54.2	**80.4**

Table 2: Part segmentation results on ShapeNet Part. The number appending IT-Net indicates the number of unfolded iterations during training. The metric is mIoU(%) on points and the base segmentation model is DGCNN [5].

Classifier	PointNet						
Transformer	None	T-Net		T-Net reg		IT-Net (ours)	
# Iterations	0	1	2	1	2	1	2
Accuracy	59.97	66.04	35.13	65.84	67.06	68.72	**69.94**
Classifier	DGCNN						
Transformer	None	T-Net		T-Net reg		IT-Net (ours)	
# Iterations	0	1	2	1	2	1	2
Accuracy	65.60	70.38	16.61	71.15	72.69	72.57	**74.15**

Table 3: Classification accuracy on Partial ModelNet40.

Classifier	PointNet						
Transformer	None	T-Net		T-Net reg		IT-Net (ours)	
# Iterations	0	1	2	1	2	1	2
Accuracy	62.11	63.09	30.86	62.99	61.82	63.67	**66.02**
Classifier	DGCNN						
Transformer	None	T-Net		T-Net reg		IT-Net (ours)	
# Iterations	0	1	2	1	2	1	2
Accuracy	66.02	72.75	18.55	74.12	70.80	76.36	**76.66**

Table 4: Classification accuracy on ScanNet Objects.

3.2. 3D Shape Classification

We compare three 3D transformers, IT-Net, T-Net [4] and T-Net reg by training them jointly with state-of-the-art shape classification networks [4, 5]. The transformers share the same architecture except for the last output layer. IT-Net outputs 7 numbers for rotation (quaternion) and translation; T-Net and T-Net reg outputs 9 numbers to form a 3×3 affine transformation matrix A. For T-Net reg, a regularization term $\|AA^T - I\|$ is added to the loss with weight 0.001. Each transformer is unfolded for up to 2 iterations using the iterative scheme described in Sec. 2.2.

Table 3 and 4 show the classification accuracy on Partial ModelNet40 and ScanNet Objects respectively. It can be seen that IT-Net consistently outperforms other transformers when trained with different classifiers. Further, the advantage of IT-Net on real data (Table 4 matches that on synthetic data (Table 3). This demonstrates IT-Net's ability to process point clouds with realistic sensor noise and clutter.

3.3. Object Part Segmentation

To show that the advantage of IT-Net is not specific to classification, we replaced the classification network in Sec. 3.2 with a segmentation network and evaluated the part segmentation performance on ShapeNet Part. We use the mean Intersection-over-Union (mIoU) on points as the evaluation metric following [4, 5]. Our results in Table 2 show that IT-Net outperforms baselines in terms of mean mIoU and mIoU for most categories. This demonstrates the potential of IT-Net as a plug-in module for any task that requires invariance to geometric transformations without task-specific adjustments to the model architecture.

4. Conclusion

In this work, we propose Iterative Transformer Network (IT-Net), a novel 3D transformer network that outputs a rigid transformation in an iterative fashion. IT-Net can be used to accurately estimate object pose or improve the performance of existing point-based shape classification and segmentation networks on partial, misaligned inputs.

References

[1] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 1, 3

[2] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 3

[3] M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. In *Advances in neural information processing systems*, pages 2017–2025, 2015. 1

[4] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 1(2):4, 2017. 1, 2, 3, 4

[5] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon. Dynamic graph cnn for learning on point clouds. *arXiv preprint arXiv:1801.07829*, 2018. 1, 4

[6] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015. 1, 3