Point-JEPA: A Joint Embedding Predictive Architecture for Self-Supervised Learning on Point Cloud

Supplementary Material

A. Further Pre-training Details

Optimization We utilize AdamW [3] optimizer with cosine learning decay [2]. Starting from learning rate of 10^{-5}, we increase it to 10^{-3} in the first 30 epochs and decay it to 10^{-6}. The batch size for pretraining is set to 512, and β for Smooth L1 loss is set to 2, similar to Point2Vec [6]. The target encoder and context encoder initially have identical parameters. The context encoder's parameters are updated via backpropagation, while the target encoders' parameters are updated using the exponential moving average of the context encoder parameters, that is $\overline{\theta} \leftarrow \tau\overline{\theta} + (1-\tau)\theta$ where $\tau \in [0, 1]$ denotes the decay rate. We gradually increase the decay rate of the exponential moving average from 0.995 to 1.0 during pretraining.

Masking and Ordering To determine the sequence of patch embeddings, we utilize the iterative ordering of associated center points, as previously mentioned. We chose the starting point in this sequence with the lowest sum of its coordinates. This method allows us to start the sequence from a point on the outer edge of the object rather than from a point within the object's interior. This consistency in selecting the initial point is experimentally shown to deliver a slightly better learned representation than taking the first available index.

For masking, we define a range of ratios with both upper and lower limits similar to I-JEPA [1]. To start with, we clarify that the term "block" refers to a sequence of patch embeddings and their corresponding encoded embeddings that are contiguous. Because of the sequencing process applied before the target and context selection, most contiguous patch embeddings and encoded embeddings are also spatially contiguous. For the target, we randomly select 4 blocks of encoded embeddings processed by transformer blocks from within the 0.15 to 0.2 range. We then remove the corresponding patch embeddings of encoded embedding vectors that have already been chosen as targets for further selection. Following this, we choose a block of patch embeddings that is within the range of 0.4 to 0.75 out of available patch embeddings that are not concealed. Because some of the patch embeddings are not available for context selection, we note that context block usually consists of multiple sets of patch embeddings that are spatially contiguous. The selection of targets is completed on a per-batch basis, and we track the indices of these targets to ensure that the corresponding patch embeddings of these selected encoded embeddings are concealed in the context selection. The context is then selected using the available indices of patch embeddings also on a per-batch basis.

Targets		**Context**	**OA**
Ratio	Freq.	Ratio	Modelnet40 Linear
(0.1, 0.2)	4	(0.85, 1.0)	93.0
(0.15, 0.2)	4	(0.85, 1.0)	**93.3**
(0.2, 0.25)	4	(0.85, 1.0)	93.2
(0.25, 0.3)	4	(0.85, 1.0)	92.4
(0.3, 0.35)	4	(0.85, 1.0)	90.5
(0.35, 0.4)	4	(0.85, 1.0)	84.6

Table 1. **Ratio Range for Target.** The ratio of encoded embedding vectors selected for each target.

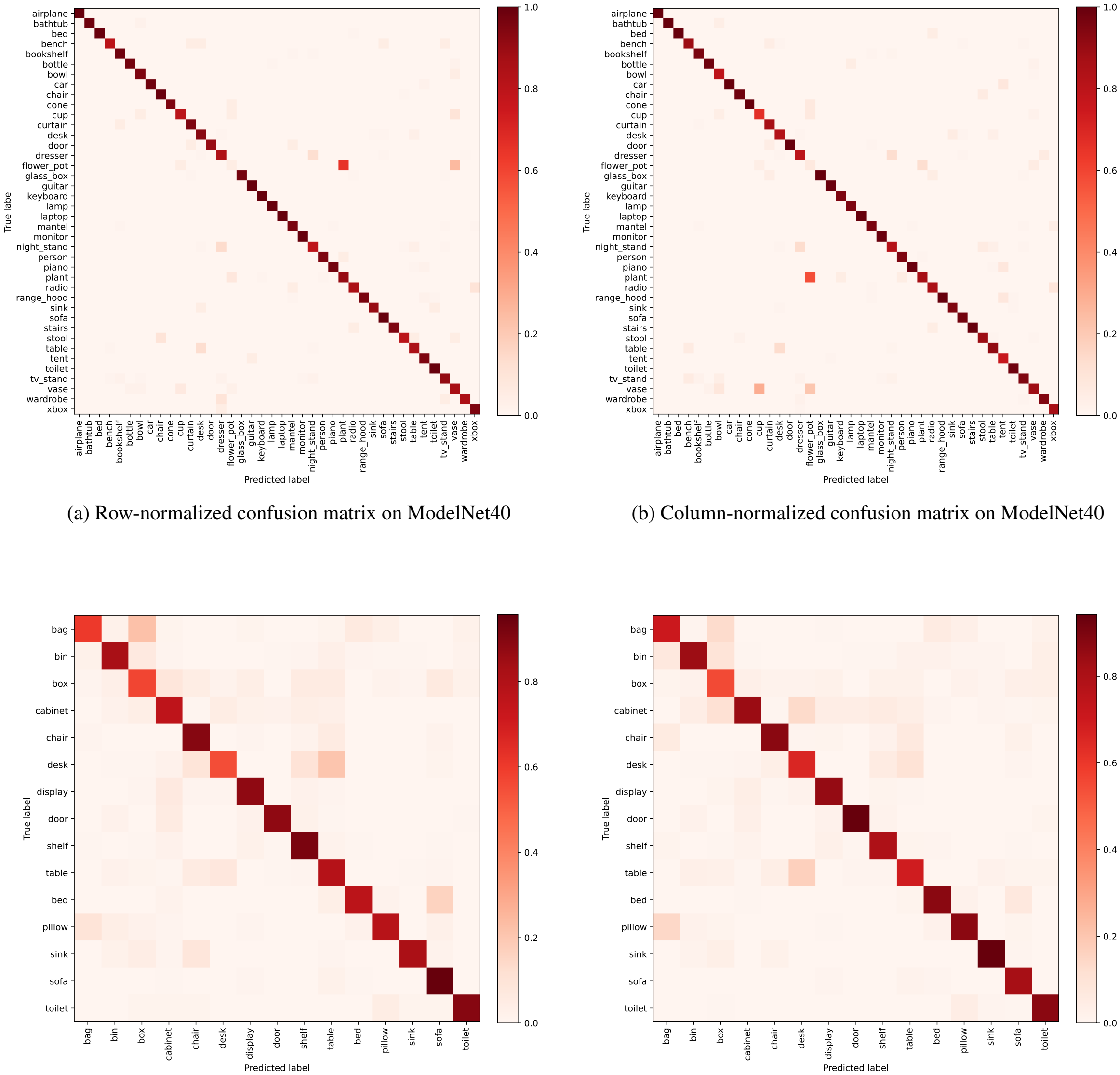
B. Further Ablation

Ratio of Targets. We change the ratio of the selected embedding vectors for the target selection while keeping the number of target blocks and the ratio of context patch embedding fixed. As shown in Tab. 1, the performance increases when you increase the ratio to a certain point. However, beyond this point, further increasing the ratio results in decreased performance. This implies that Point-JEPA does not require a large size for the target blocks and benefits from a sufficient amount of available patch embeddings for context selection.

Targets		**Context**	**OA**
Ratio	Freq.	Ratio	Modelnet40 Linear
(0.15, 0.2)	4	(0.85, 1.0)	93.1
(0.15, 0.2)	4	(0.75, 1.0)	92.8
(0.15, 0.2)	4	(0.65, 1.0)	93.4
(0.15, 0.2)	4	(0.45, 1.0)	93.6
(0.15, 0.2)	4	(0.6, 0.75)	93.4
(0.15, 0.2)	4	(0.5, 0.75)	93.1
(0.15, 0.2)	4	(0.4, 0.75)	**93.7**

Table 2. **Ratio Range for Context.** The ratio of patch embeddings selected for context encoding.

Ratio of Context. In this study, we change the ratio of patch embeddings selected for context encoding while keeping the number of targets and the ratio range for targets



(a) Row-normalized confusion matrix on ModelNet40

(b) Column-normalized confusion matrix on ModelNet40

(c) Row-normalized confusion matrix on ScanObjNN

(d) Column-normalized confusion matrix on ScanObjNN

Figure 1. Confusion matrices illustrating model performance on ModelNet40 and another dataset, highlighting class-specific accuracies and challenges with similar categories.

fixed. As shown in Table 2, having a relatively large difference between the lower and upper bound of the ratio can improve performance. In other words, Point-JEPA learns a better representation when the number of selected context patch embeddings varies more between training iterations. Additionally, when the upper bound of the ratio is somewhat constrained, we see increased performance.

Predictor Depth We also study the effect of the predictor's depth on the learned representation. To this end, we vary the predictor depth and observe its effect on the linear evaluation accuracy. As shown in Table 3, Point-JEPA benefits from a deeper predictor.

Class confusion on ModelNet40 and ScanObjNN To assess our model's performance on the ModelNet40 [5] and ScanObjNN [4] datasets, we present two types of visualiza-

Predictor Depth	Modelnet40 Linear (OA)
2	92.5
3	92.8
4	93.2
5	93.4
6	**93.7**

Table 3. **Predictor Depth.** Predictor depth and its effect on learned representation.

tions for each dataset. The first is a row-normalized confusion matrix, which illustrates the model's sensitivity, indicating how well the model identifies each actual class. The second is a column-normalized confusion matrix, depicting the model's specificity, which shows the correctness of predictions for each class assumed by the model. As illustrated in parts (a) and (b) of Fig. 1, the model fine-tuned on ModelNet40 demonstrates high accuracy. At the same time, errors predominantly arise from similar categories within the dataset. For instance, "flower pot" and "plant" are often misclassified, likely due to the presence of flowers in some of the flower pot models in the ModelNet40 dataset. Similarly, parts (c) and (d) of Fig. 1 show the aforementioned confusion matrices. As highlighted in the main paper, our model's performance on ScanObjectNN dataset has room for enhancement compared to ModelNet40. The confusion matrix reveals some misclassifications, but it is encouraging to see that these errors predominantly occur between closely related classes, such as 'table' and 'desk' or 'sofa' and 'bed'. This suggests that our model has a solid grasp of the key characteristics of these categories and that further refinement of the classification criteria could lead to significant improvements in overall accuracy.

References

[1] Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15619–15629, 2023. 1

[2] Ilya Loshchilov and Frank Hutter. SGDR: stochastic gradient descent with restarts. *CoRR*, abs/1608.03983, 2016. 1

[3] Ilya Loshchilov and Frank Hutter. Fixing weight decay regularization in adam. *CoRR*, abs/1711.05101, 2017. 1

[4] Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Duc Thanh Nguyen, and Sai-Kit Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. *CoRR*, abs/1908.04616, 2019. 2

[5] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1912–1920, 2015. 2

[6] Karim Abou Zeid, Jonas Schult, Alexander Hermans, and Bastian Leibe. Point2vec for self-supervised representation learning on point clouds. In *DAGM German Conference on Pattern Recognition*, pages 131–146. Springer, 2023. 1