VIBUS: Data-efficient 3D Scene Parsing with Viewpoint Bottleneck and Uncertainty-Spectrum Modeling

Beiwen Tian\textsuperscript{a,e,*}, Liyi Luo\textsuperscript{b,e,*}, Hao Zhao\textsuperscript{c,d}, Guyue Zhou\textsuperscript{e,**}

\textsuperscript{a}Department of Computer Science and Technology, Tsinghua University, Beijing, China
\textsuperscript{b}Bioresource Engineering Department, McGill University, Montreal, Canada
\textsuperscript{c}Intel Lab, Beijing, China
\textsuperscript{d}Peking University, Beijing, China
\textsuperscript{e}Institute for AI Industry Research, Tsinghua University, Beijing, China

Abstract

Recently, 3D scenes parsing with deep learning approaches has been a heating topic. However, current methods with fully-supervised models require manually annotated point-wise supervision which is extremely user-unfriendly and time-consuming to obtain. As such, training 3D scene parsing models with sparse supervision is an intriguing alternative. We term this task as data-efficient 3D scene parsing and propose an effective two-stage framework named VIBUS to resolve it by exploiting the enormous unlabeled points. In the first stage, we perform self-supervised representation learning on unlabeled points with the proposed Viewpoint Bottleneck loss function. The loss function is derived from an information bottleneck objective imposed on scenes under different viewpoints, making the process of representation learning free of degradation and sampling. In the second stage, pseudo labels are harvested from the sparse labels based on uncertainty-spectrum modeling. By combining data-driven uncertainty measures and 3D mesh spectrum measures (derived from normal directions and geodesic distances), a robust local affinity metric is obtained. Finite

\textsuperscript{*}These authors contributed equally to this work.
\textsuperscript{**}Corresponding author

Email addresses: tbw18@mails.tsinghua.edu.cn (Beiwen Tian), liyi.luo@mail.mcgill.ca (Liyi Luo), hao.zhao@intel.com (Hao Zhao), zhao-hao@pku.edu.cn (Hao Zhao), zhouguyue@tsinghua.edu.cn (Guyue Zhou)
gamma/beta mixture models are used to decompose category-wise distributions of these measures, leading to automatic selection of thresholds. We evaluate VIBUS on the public benchmark ScanNet and achieve state-of-the-art results on both validation set and online test server. Ablation studies show that both Viewpoint Bottleneck and uncertainty-spectrum modeling bring significant improvements. Codes and models are publicly available at https://github.com/AIRDISCOVER/VIBUS.

**Keywords:** 3D scene understanding, self-supervised learning, weakly-supervised representation learning, uncertainty analysis, spectral clustering

**2010 MSC:** 00-01, 99-00

1. Introduction

As a crucial component of 3D computer vision, 3D scene parsing provides the ability of fine-grained understanding for various downstream tasks from robotic manipulation [1], indoor navigation [2] [3] to geographic information system analysis [4]. Various deep-learning-based methods, including graph convolutional network [5], neural architecture search [6] and joint residual-dense optimization [7] have been proposed to address this task in a fully-supervised manner. However, point-wise annotation of large-scale 3D scenes is very time-consuming [8] [9]. Some works propose to address this issue by involving Virtual Reality [10] [11], Augmented Reality [12] [13] or other interactive systems [14], in which the procedures for labeling are more user-friendly but still inevitable and time-consuming. To this end, training 3D scene parsing models with only annotations on sparse points becomes appealing, since annotating all points in a point cloud is not practical in industry. We call this task data-efficient 3D scene parsing and propose a method named VIBUS to address it. Fig. 1-(e) shows the parsing result generated by a model trained using 200 points per scene, which is almost as good as the fully supervised result shown in Fig. 1-(f). Since a scene in the ScanNet dataset [15] typically contains more than 50,000 points, a subset
Figure 1: We address the challenging problem of learning 3D scene parsing models using only semantic annotations on limited points (20 to 200 points per scene). (a) shows a point cloud of an indoor scene. (b)-(e) demonstrate 3D semantic understanding results using 20 to 200 point annotations per scene. (f) shows the result generated by a fully supervised model. Except for highlighted regions in boxes, most points are correctly segmented.

of 200 points only amounts to 0.4% data usage.

The key to data-efficient 3D scene parsing is leveraging numerous unlabeled points (e.g., the other 99.6% data in the aforementioned case). There are two natural ideas to achieve this goal: (1) If we can learn meaningful representations from unlabeled points without the usage of semantic annotation, a subsequent data-efficient fine-tuning step is expected to give good results. This is usually referred to as self-supervised representation learning (SSRL). (2) Models trained with sparse annotations can generate pseudo labels for unlabeled points in the training set. If we can harvest correct pseudo labels and fine-tune the model with them, performance is expected to be boosted. However, applying these two ideas to 3D scenes is challenged by several issues:

**Challenges on SSRL.** While exciting progress has been achieved in 2D SSRL [16][17][18], 3D SSRL in point clouds [19][20] is still an under-explored emerging topic. Several open problems exist and challenge the effectiveness of
existing 3D SSRL methods: (1) Former 2D SSRL methods treat each image as a sample, whereas 3D SSRL for dense semantic parsing naturally requires us to treat each point as a sample. This fact leads to an extremely large sample set. (2) Former contrastive learning methods operate on the sample dimension. As conceptually shown in the left part of Fig. 3-(b), embeddings for the same sample under different viewpoints are drawn near, and embeddings for different samples are pushed away. Selecting good sample pairs is an unclear and difficult problem, especially when the 3D sample set is large. (3) Former contrastive learning methods are troubled with degenerated solutions thus require sophisticated techniques to break the symmetry like weight averaging with momentum [18]. Complex implementation details make them hard to tune.

**Challenges on pseudo labels.** The most straightforward way to select pseudo labels is thresholding softmax scores. However, it is known that softmax scores can be over-confident on false predictions [21] and vulnerable to adversarial attacks [22]. Another natural idea is to select pseudo labels that are geometrically close to labeled points, but defining reliable affinity metrics on 3D scene meshes is more difficult than 2D images. For example, two vertices can be close to each other in the Euclidean space but belong to different categories. Lastly, no matter which metric is used, tuning thresholds harms the generalization ability of an algorithm.

The proposed method leverages both ideas: an objective called VI**ewpoint B**ottleneck allows effective 3D SSRL and a module based upon U**ncertainty-S**pectrum modeling generates reliable pseudo labels. As such, the method is shortened as VIBUS. VIBUS addresses aforementioned challenges: (1) Several recent SSRL methods [23][24][25] switch representation learning from the sample dimension to the feature dimension. We introduce this idea into 3D SSRL, as conceptually shown in the right part of Fig. 3-(b). There is no need to select sample pairs or design sophisticated techniques that break symmetry, leading to a simple implementation. (2) Other than softmax scores, we use Monte Carlo Dropout [26] to generate uncertainty measures that reliably reflect pseudo label quality. (3) Geodesic distance and normal directions are combined into
a unified geometric embedding via spectral analysis, which serves as principled
local affinity measures for 3D meshes. (4) Gamma or beta mixture models are
used to decompose uncertainty and spectrum measures so that thresholds can
be automatically selected with the mixture distribution.

Our method is empirically successful, achieving state-of-the-art results on
the validation set and test server of a widely used benchmark named ScanNet.
Ablations show that both Viewpoint Bottleneck pre-training and uncertainty-
spectrum modeling lead to clear performance margins. Furthermore, we also
design an interactive point cloud annotator based on [27] for sparse point cloud
annotations. This tool along with our methods have been widely used in
an industrial partner for their robot vacuum scene understanding cloud plat-
form. Our codes and models are publicly available at https://github.com/AIR-
DISCOVER/VIBUS.

2. Related works

2.1. Self-supervised Representation Learning
SSRL is a special form of unsupervised learning, which aims to learn meaning-
ful representations of the input data without relying on annotations. The
common ground of these methods is to learn representations that are invariant
to distortions. There are several ways to achieve this general principle. For ex-
ample, early attempts use different surrogate tasks whose labels can be naturally
generated, like pixel value inpainting [28][17], cross-channel feature regression
[29], or rotation prediction [30]. Recently, contrastive learning [31], which was
first proposed for metric learning, has drawn significant attention and witnessed
great success, such as DenseCL [32] and Alonso et al. [33]. However, it is now
clear that a trivial contrastive learning formulation suffers from degeneration.
As such, many variants have been proposed to address the issue like weight
averaging [18][34] or stop gradient [35] among others [36][37].

Compared to its 2D counterpart, 3D SSRL is more urgent because of the dif-
ficulty of annotating 3D data. There are already some pioneering works that ad-
Figure 2: The overall pipeline of our method. In the first stage (A), the backbone network is pre-trained in a self-supervised manner named Viewpoint Bottleneck. With a simple implementation, Viewpoint Bottleneck decorrelates different components of the representations and exploits the intrinsic structures of the unlabeled points. In stage (B), the backbone network is fine-tuned with sparse annotations. For each scene, only 20/50/100/200 points have ground-truth labels. With the fine-tuned model, we inference on the training dataset and obtain the predicted label for each point. In stage (C), we perform uncertainty analysis and spectrum analysis, and build double mixture models based on the statistics from the analyses. With the mixture models, unreliable labels are filtered out, and reliable labels are selected to form the pseudo label sets. In stage (D), The backbone network is fine-tuned once again with the harvested pseudo labels. In stage (E), we inference on the validation set with the fine-tuned model in stage (D).

dress the 3D SSRL problem borrowing ideas of contrastive learning, like DepthContrast [38], PointContrast [19] and Contrastive Scene Contexts (CSC) [20]. DepthContrast works with single-view depth scans that are collected by varied sensors and without any 3D registration and point correspondences. PointContrast proposes a PointInfoNCE loss and verifies its effectiveness on a diverse set of scene understanding tasks. However, it ignores the spatial context around local points, limiting its transferability to complex downstream tasks. CSC introduces a loss function that contrasts features aggregated in local partitions. It demonstrates the possibility of using extremely few annotations to achieve
high performance on a series of data-efficient settings. Liu et al. \cite{39,40} propose to use multi-modal pairs and tuples as elements for RGB-D contrastive learning and scene understanding. Huang et al. \cite{41} develops a sophisticated spatial-temporal contrastive learning framework to point clouds.

To address the common issues of contrastive learning, a recent SSRL method called Barlow Twins \cite{23} proposes an objective function that measures the cross-correlation matrix between the representations of two distorted versions of a sample. It naturally avoids representation collapse and sample selection since no negative samples are required. Bardes et al. \cite{24} and li et al. \cite{25} enrich this new scheme with a variance term and a principled kernel-based statistical measure. Inspired by them, we propose the Viewpoint Bottleneck principle, the first non-contrastive 3D SSRL method.

2.2. Uncertainty analysis in deep learning

There are many ways to obtain uncertainty measures for deep learning networks. In our method, we obtain uncertainty measures with multiple inferences with dropout layers \cite{26}. It officially demonstrates the association between the output variance of multi-layer perceptrons with dropout layers and the predictive distribution. Zhao et al. \cite{42} counts on the dropout layer before the final classifier for fully convolutional models in PSPNet. In another way, uncertainty measures can be calculated as the root square of the trace of the output covariance matrix. Jena et al. \cite{43} exploits the reparameterization trick of adding a Gumbel-distributed random variable onto the activations and dividing them by an annealing parameter. Lakshminarayanan et al. \cite{44} proposed to calculate uncertainty measures by gathering predictions from an ensemble.

2.3. Spectral clustering

Spectral clustering is one of the most widely used techniques for data analysis. Liu et al. \cite{45} firstly applied spectral clustering to 3D mesh segmentation. However, it is not easy to apply spectral clustering on large-scale point clouds. The natural way to solve this problem is using sampling methods. Shinnou
et al. [46] proposed the Committees-based Spectral Clustering, in which the committees are some points that are near to the cluster centers. They use these committees to construct the similarity matrix. Then, the spectral clustering is performed by the reduced similarity matrix. Chen et al. [47] developed the Landmark-based Spectral Clustering (LSC), in which they selected $p(\ll n)$ representation data points as the landmarks and represented the original data points as the linear combinations of these landmarks. Then the spectral embedding of the data can be computed with the landmark-based representation. In our work, we use the Quadric Error Metrics [48] to reduce the size of a large-scale point cloud. Then, the traditional spectral clustering is applied to the sampling point clouds.

3. Viewpoint Bottleneck

In this section, we describe the details of the proposed self-supervised representation learning framework named Viewpoint Bottleneck. By introducing the novel Viewpoint Bottleneck loss, we aim to exploit unlabeled points and learn meaningful representations.

3.1. Self-supervised Representation Learning Pipeline

In this section, we describe the general pipeline of Viewpoint Bottleneck.

Firstly, a point cloud $X$ is transformed into two viewpoints by random geometric transformations. Transformed 3D point clouds are denoted by $X_p$ and $X_q$, respectively. Then, the two randomly transformed point clouds are fed into the same sparse convolution network $f_\theta$ to obtain two high-dimensional representations $Z_p = f_\theta(X_p)$ and $Z_q = f_\theta(X_q)$.

Next, we sample the representations by Farthest Point Sampling (FPS) from $M_i \times D$ to $H_i \times D$, where $D$ is the dimension of the representations and $M_i$, $H_i$ are the numbers of points before and after sampling. FPS starts with a random point as the first proposal candidate and iteratively selects the farthest point from the already selected points until $H_i$ candidates are selected. We set $H_i \ll $
Figure 3: The principles of Viewpoint Bottleneck. In (a), we show that Viewpoint Bottleneck aims to maximize the mutual information between the input point cloud and the feature representation and at the same time minimize the mutual information between the transformed point cloud and the feature representation. In (b), we demonstrate that Viewpoint Bottleneck decorrelates different components of the representations. Instead of operating on the dimension (demonstrated in (b)-left) like former contrastive learning methods, Viewpoint Bottleneck operates on the feature dimension (demonstrated in (b)-right), leading to a simple pipeline free of degradation and sampling.

After FPS, the cross-correlation matrix $Z(D \times D)$ is computed on the batch dimension, from the two sampled representations $Z_q'$ and $Z_q''$. Finally, the network is trained with the proposed Viewpoint Bottleneck loss, as detailed later.

### 3.2. Viewpoint Bottleneck Loss

Inspired by Barlow Twins [23], we derive our Viewpoint Bottleneck objective from an information theory perspective [49][50]. As shown in Fig. 3, the information bottleneck principle assumes that meaningful representations should
reserve as much information as possible about the input while affected by random viewpoint transformations as little as possible. Formally the objective is stated as:

$$\text{VB}_\theta \triangleq I(C_\theta, B) - \beta I(C_\theta, A)$$  \hspace{1cm} (1)$$

where $I(\cdot, \cdot)$ is mutual information between two random vectors. Here $A$, $B$, and $C_\theta$ correspond to input point clouds, randomly transformed point clouds, and learned representations, respectively. $\beta$ is a balancing weight. By definition, the mutual information can be calculated as $I(C_\theta, B) = H(C_\theta) - H(C_\theta|B)$ (similarly for $I(C_\theta, A)$). $H(C_\theta)$ is the entropy of $C_\theta$ while $H(C_\theta|B)$ is the conditional entropy of $C_\theta$ given $B$. As the function $f_\theta$ is deterministic, $H(C_\theta|B)$ cancels to 0. We rewrite the Eq.1 as:

$$\text{VB}_\theta = H(C_\theta) - \beta [H(C_\theta) - H(C_\theta|A)]$$  \hspace{1cm} (2)$$

By dividing Eq.2 with $\beta$, we have:

$$\text{VB}_\theta = H(C_\theta|A) + \frac{1 - \beta}{\beta} H(C_\theta)$$  \hspace{1cm} (3)$$

However, the optimization of $\text{VB}_\theta$ becomes a challenge as evaluating the entropy of a generic random vector is intractable. We hereby derive a surrogate loss. We assume the representation $\vec{C}_\theta$ is distributed as a Gaussian: $\vec{C}_\theta = (c_{\theta 1}, c_{\theta 2}, \cdots, c_{\theta D}) \sim \mathcal{N}(\mu, \Sigma_{C_\theta})$, where $D$ is the dimension of the representation.

$\Sigma_{C_\theta}$ can be decomposed into $\Sigma_{C_\theta} = E^T E$, and $Y = E^{-1}(\vec{C}_\theta - \vec{\mu}) \sim \mathcal{N}(\vec{0}, I)$. In the following derivation, $|\cdot|$ means taking determinant. The probability density function of $\vec{C}_\theta$ is:

$$p(c_{\theta 1}, c_{\theta 2}, \cdots, c_{\theta D}) = \frac{1}{(2\pi)^{\frac{D}{2}} |\Sigma_{C_\theta}|^\frac{1}{2}} e^{-\frac{1}{2} [(c_{\theta} - \vec{\mu})^T \Sigma_{C_\theta}^{-1} (c_{\theta} - \vec{\mu})]}$$  \hspace{1cm} (4)$$

where $D$ is the dimension of $\vec{C}_\theta$. 

10
Then according to \cite{51}:

\[
H(C_\theta) = -\int p(c_\theta) \log p(c_\theta) \, dc_\theta \\
= -\int p(c_\theta) \log e^{-\frac{1}{2}(\vec{C}_\theta - \vec{\mu})^T \Sigma^{-1}_C (\vec{C}_\theta - \vec{\mu})} \\
= -\int p(c_\theta) \log \frac{1}{(2\pi)^\frac{D}{2} |\Sigma_C|^\frac{1}{2}} dc_\theta \\
- \int p(c_\theta) \log e^{-\frac{1}{2}(\vec{C}_\theta - \vec{\mu})^T (E^{-1})^T E^{-1}(\vec{C}_\theta - \vec{\mu})} dc_\theta \\
= \log (2\pi)^\frac{D}{2} |\Sigma_C|^{\frac{1}{2}} - \int p(Y) \log e^{-\frac{1}{2}Y^T Y} dy \\
= \log (2\pi)^\frac{D}{2} |\Sigma_C|^{\frac{1}{2}} - \sum_{i=1}^{D} \log e^{-\frac{1}{2}|y_i^2|} \\
= \log (2\pi)^\frac{D}{2} |\Sigma_C|^{\frac{1}{2}} + \log e^{\frac{D}{2}} \\
= \frac{1}{2} \log (2\pi e)^D |\Sigma_C| \\
\]

(5)

As such the Viewpoint Bottleneck objective Eq.3 is turned into a practical surrogate loss function as:

\[
H(C_\theta|A) = \sum_a p(a) H(C_\theta|A = a) \\
= \frac{1}{2} E_A \log (2\pi e)^n |\Sigma_C| |A| \\
\]

(6)

Finally, we make several more simplifications: 1) Since Eq.7 is only meaningful when \(\beta > 1\), we use a positive constant \(\lambda\) to replace \(\frac{\beta - 1}{\beta}\). 2) In the second term of the Eq.7, we use the Frobenius norm of \(\Sigma_C\) as the metric. 3) Recall that the first term of Eq.7 is derived from the conditional entropy of the representation \(C_\theta\) given \(A\). Minimizing it is equivalent to maximizing the diagonal elements of \(Z\) (the aforementioned \(D \times D\) cross-correlation matrix).
3.3. Viewpoint Bottleneck: Implementation

After making aforementioned three simplifications to Eq.7, we obtain the final Viewpoint Bottleneck loss function as:

$$\mathcal{L}_{VB} \triangleq \| \Gamma_{\lambda}(Z) - I \|_F$$  \hspace{1cm} (8)

where $I$ denotes the identity matrix, $\Gamma_{\lambda}(\cdot)$ denotes scaling off-diagonal elements by positive constant $\lambda$, and $Z$ is the cross-correlation matrix computed between the two representations of different viewpoints along the feature dimension:

$$Z \triangleq \left( \tilde{Z}_p \right)^T \tilde{Z}_q$$  \hspace{1cm} (9)

$\tilde{Z}'(H \times D)$ is the representation obtained by normalizing $Z'$ along the sample dimension.

$\mathcal{L}_{VB}$ aims to push the diagonal elements of $Z$ towards 1, so that the representations are invariant to random geometric transformations. Meanwhile, by forcing the off-diagonal elements towards 0, we decorrelate different vector components of the representations that accordingly contain non-redundant information about the original point cloud.

Without complex network structures (e.g., asymmetric network, weight averaging), the intrinsic structures between enormous unlabeled points are fully leveraged by the training scheme of Viewpoint Bottleneck, making it an efficient way to learn effective representations from the unlabeled points.

The pipeline of Viewpoint Bottleneck is also demonstrated in Algorithm. We then fine-tune the network using labels and pseudo labels harvested by the methods that we introduce in Section.
Algorithm 1: 3D SSRL with Viewpoint Bottleneck

Notations:

- \( X \): a set of 3D point clouds;
- \( T \): the distribution of geometric transformations;
- \( f_\theta \): sparse ConvUNet parameterized by \( \theta \);
- \( K \): the number of optimization steps;
- \( \lambda \): the control parameter in VB loss
- FPS: Farthest Point Sampling, takes a point cloud as input and outputs the indices of sampled points

for \( k = 1 \) to \( K \) do

    /* sample transformations */
    \( T_p, T_q \sim T \)

    /* take an input point cloud */
    \( X \in X \)

    /* apply transformations */
    \( X_p = T_p(X), X_q = T_q(X) \)

    /* extract features */
    \( Z_p = f_\theta_k(X_p), Z_q = f_\theta_k(X_q) \)

    /* perform farthest point sampling */
    /* on the original point clouds */

    /* use the sampled indices to select features */
    \( Z'_p = Z_p[\text{FPS}(X_p)], Z'_q = Z_q[\text{FPS}(X_q)] \)

    /* calculate cross-correlation */
    \( Z = (Z'_p)^T Z'_q \)

    /* calculate VB loss */
    \( L_{\text{VB}} = \| \Gamma_\lambda(Z) - T \|_F \)

    /* optimize network parameters */
    \( \theta_{k+1} = \text{optim}(\theta_k, L_{\text{VB}}) \)

end
Figure 4: Illustration of the harvesting process of pseudo labels. To perform uncertainty analysis, we repeatedly inference with the pre-trained model with effective dropout layers and calculate the variances of the digits as the uncertainty of the predicted labels. To perform spectral analysis, we perform k-means clustering with the combination of geodesic and angular distance as the distance measure. The spectrum distance of a point to its cluster center is regarded as the spectrum distance of the associating predicted label. With statistics obtained from the analyses, we fit double mixture models for each category. Each double mixture model consists of a reliable and an unreliable component. The pseudo labels set are then generated by keeping the predicted labels that have a larger probability to be drawn from the reliable part.

4. Uncertainty-Spectrum Analysis and Pseudo Label Harvesting

To address the issue of pseudo label harvesting, we seek to fully exploit the labeled points to generate new labels for the unlabeled points. To achieve this goal, we first fine-tune the pre-trained model with the sparse annotations and obtain the predictions of categories for each point in the scene. Then the predictions are then filtered so that predictions with high confidence are collected as pseudo labels, which we regard as the supervision for the second fine-tuning process.

In the following sections, we propose three strategies for the aforementioned filtering process: uncertainty modeling, spectrum modeling, and uncertainty-spectrum joint modeling.
4.1. Uncertainty Modeling

When a neural network generates deterministic predictions, it can be problematic that the model is overconfident with wrong predictions, since deterministic results cannot show the model confidence. To alleviate this situation, a better practice is to generate the predictions with probability distributions.

One possible approach is to treat the network parameters as random variables, but iterating all possible parameters of a neural network is unrealistic. Gal et al. put forward a workaround to assess the uncertainty of deep neural networks by running the model with functional dropout layers for multiple rounds and calculating the variances of the logits. Recent work by Zhao et al. argues that dropout-based uncertainty is a valid criterion for label quality and achieves state-of-the-art performances of semantic segmentation on real-world 2D image datasets by applying the criterion to select reliable pseudo labels.

Following this insight, we aim to leverage the class-wise dropout-based uncertainty of pseudo labels in 3D scenes, which was not well studied before this work. With dropout layers functioning in the pre-trained network, we run inferences on the training set multiple times and calculate the variance of digits in multiple runs as the uncertainty of the predictions.

We show visualized uncertainty in Fig. 5. We observe that points with incorrect predictions tend to have higher uncertainty and vice versa, as shown in Fig. 6. Furthermore, the frequency histograms show non-trivial patterns that are similar with [52].

Following this keen observation and also inspired by [52], we propose a double mixture model to fit the distribution of uncertainty in a class-wise manner. We assume that the uncertainty distribution consists of a certain and an uncertain part and use the two components of the mixture model to fit these two parts, respectively. We adopt gamma distribution as the probability density function, since the uncertainty is intrinsically variance thus always positive and the distribution is long-tailed in our observations. Therefore, the mathematical
formulation of the probability density of the uncertainty being $x$ is given by

$$P(x|\theta_1, \theta_2) = \omega_1 \cdot P(x|\theta_1) + \omega_2 \cdot P(x|\theta_2)$$

$$P(x|\theta) = \frac{b^a x^{a-1} e^{-bx}}{\Gamma(a)}, \quad \theta = \{a, b\}$$

where $\omega_1$ is the probability that a sample is drawn from the certain part and $\omega_2$ is the probability that a sample is drawn from the uncertain part. $\theta_1$ and $\theta_2$ are the parameter sets of the two components. Each parameter set $\theta$ consists of two independent parameters $a$ and $b$ while $a$ is the shape parameter that controls the general shape of the distribution and $b$ is the scale parameter demonstrating the degree of concentration.

The detailed math derivation of fitting double mixture models is described in Sec. 4.4. With the double mixture model, we select from the generated labels and harvest reliable pseudo labels that are more likely to belong to the certain part than the uncertain part.

Figure 5: The visualization of uncertainty. Yellow parts have relatively higher uncertainty than blue parts.

4.2. Spectrum Modeling

Intuitively, local areas around points with ground-truth labels tend to share the same semantic labels. However, in uncertainty modeling, this empirical fact is overlooked. In this section, we seek to apply clustering methods on the
dataset to further utilize the locality of semantic labels. To fully exploit the mesh connectivity information in the dataset, we choose to perform clustering based on spectrum distances rather than Euclidean distances, since spectral analysis is able to entangle different measures into a uniform feature space.

4.2.1. Distance Metric

To perform spectral clustering, the metric to evaluate the locality of points is crucial, and spatial distance between points is a crucial measurement of the locality.

However, Euclidean distance fails to play this role as adjacent points in Euclidean space may differ dramatically in semantics due to the absence of mesh edge connection. To fully exploit the relations between points from mesh edges, we adopt geodesic distance to measure the spatial distance between points.

Instead of calculating in a brute-force manner, we estimate geodesic distances along mesh edges by heat diffusion method [53] which is much faster and reasonably accurate. In this method, the shortest distances along meshes between one point \( v \) and all other points are estimated by simulation of heat diffusion along mesh edges starting at \( v \). Specifically, the geodesic distance between point \( x \) and \( y \) is derived as:

\[
d_g(x, y) = \lim_{t \to 0} \sqrt{-4t \log h_{t,x}(y)}
\]  

(11)
where $h_{t,x}(y)$ is the heat kernel measuring the number of particles transferred from a source point $x$ to a target point $y$ after a short period $t$. Fig. 7(d) illustrates the geodesic distances between one point and all other points in a 3D scene.

On the other hand, points that belong to orthogonal planes tend to have inconsistent semantic labels, thus low locality. Points near plane intersections may have different semantic labels. However, with the geodesic metric, they are likely to be grouped into the same cluster due to the small geodesic distances. To address this issue, we use the angular distance $d_a$ as an additional metric to measure the extent to which the spatial neighborhoods of points are aligned.

To calculate the angular distances between two points, we first fit two closest planes of the two points and their 10 nearest neighbors. The normal vectors of the two planes are considered as the normal vectors of the two points. We then calculate the angular distance between the two points as:

$$d_a(x,y) = 1 - |\vec{n}_x \cdot \vec{n}_y|$$ (12)

where $\vec{n}_x$ and $\vec{n}_y$ denote the normal vector of two points. Fig 7(e) illustrates the normal vector of each points in a scene.

Combining the two metrics with a proportion $\delta$, we have the final distance metric between points that can be used in clustering:

$$D(x,y) = \delta \cdot \frac{d_g(x,y)}{\text{average}(d_g)} + (1 - \delta) \cdot \frac{d_a(x,y)}{\text{average}(d_a)}$$ (13)

4.2.2. Spectral Clustering

In this section, we introduce the subsequent workflow of spectral clustering performed with the proposed distance metric.

Firstly, to reduce calculation complexity, we simplify the 3D mesh with Quadric Edge Collapse Decimation \[48][54]. Then, with the proposed distance metric, we calculate a distance matrix (denoted as $D$) which is then smoothed by an exponential kernel and normalized. The resulting matrix is the affinity
Figure 7: The visualization of spectral analysis. (a) An input 3D mesh (b) The 3D mesh simplified with Quadric Edge Collapse Decimation [48] [54]. (c) The ground truth semantic visualization of all vertices. (d) Visualization of geodesic distances from a vertex to the other vertices in the scene. (e) Visualization of normal vectors of the vertices. (f) Visualization of the clustering results.

Matrix of the scene (denoted as $\tilde{Z}$):

$$
\begin{align*}
D &= (d_{ij})_{N \times N}, \quad d_{ij} = D(i, j) \\
Z &= (z_{ij})_{N \times N}, \quad z_{ij} = e^{-d_{ij}/2\sigma^2} \\
W &= \text{diag}\{\sum_i z_{i1}, \sum_i z_{i2}, \ldots, \sum_i z_{iN}\} \\
\tilde{Z} &= W^{-\frac{1}{2}} Z W^{-\frac{1}{2}} \\
\end{align*}
$$

Here $N$ is the number of points in the scene, and $\sigma$ is a parameter for smoothing. In our experiments, we take $\sigma = \frac{1}{N^2} \sum_{i,j} D(i, j)$.

Then, we compute $k$ eigenvectors of $\tilde{Z}$ that have largest eigenvalues, organize them as a matrix $V$ and normalize each row of $V$ to obtain $\tilde{V}$:

$$
\begin{align*}
V &= [v_1, v_2, \ldots, v_k] = [a_1, a_2, \ldots, a_n]^T \\
\tilde{V} &= [\tilde{a}_1, \tilde{a}_2, \ldots, \tilde{a}_n]^T \\
\end{align*}
$$

Each row of $\tilde{V}$ is regarded as the embeddings of the points.
By taking the embeddings of points with ground-truth labels as initial cluster centers, we perform K-means clustering on the embeddings and obtain the spectrum distances between each point and its cluster center. We then map these distances to the point cloud before downsampling. The complete pipeline of Spectral Clustering is also demonstrated in Algorithm 2.

We call a point is localized when its spectrum distance to the cluster center is relatively small, as smaller spectrum distances imply more semantic locality information. In order to effectively select localized pseudo labels with the spectral statistics we harvest, we propose a double mixture probability model similar to Section 4.1. For the two individual components we choose beta distribution instead, since the normalized spectrum distances are distributed more evenly in [0, 1], and the two components represent the localized and unlocalized parts, respectively. The probability density of the spectrum distance being \( x \) is then given by:

\[
P(x|\theta_1, \theta_2) = \omega_1 \cdot P(x|\theta_1) + \omega_2 \cdot P(x|\theta_2)
\]

\[
P(x|\theta) = \frac{\Gamma(a + b)}{\Gamma(a)\Gamma(b)} x^{a-1}(1 - x)^{b-1}, \quad \theta = \{a, b\}
\]

Here, \( \omega_1 \) is the probability that a sample is drawn from the localized component and \( \omega_2 \) is the probability that a sample is drawn from the unlocalized component. \( \theta_1 \) and \( \theta_2 \) are the parameter sets of the two components. Each parameter set \( \theta \) consists of two independent parameters \( a \) and \( b \) controlling the shape of the distribution.

We empirically choose the beta distribution to model the two components of the mixtures, as we observe non-zero skewness in the distribution of spectrum distances. Similar to Section 4.1, we fit a double mixture model for each category. Then, for each generated label, we calculate the probabilities that the label is drawn from the localized and unlocalized components, and regard it as a pseudo label if the former is higher than the latter.

The detailed derivation of fitting the double mixture model is provided in Sec. 4.4.
Algorithm 2: Pipeline for Spectral Analysis

Input:

1 $\mathbb{X}$: a large-scale 3D mesh dataset
2 $\mathcal{T}(\cdot)$: Quadric Edge Collapse Decimation
3 $\mathcal{K}(\cdot)$: K-means clustering
4 $\mathcal{M}(\cdot)$: Data mapping by nearest neighbor

Output: $\{d_i\}$: the distance between each point and its cluster center in spectral space

foreach $X \in \mathbb{X}$ do

/* 3D mesh simplification */
$X' = \mathcal{T}(X)$

/* Distances between points */
$D(x, y) = \delta \cdot \frac{d_g(x, y)}{\text{average}(d_g)} + (1 - \delta) \cdot \frac{d_a(x, y)}{\text{average}(d_a)}$

/* Affinity matrix */
$Z = (z_{ij})_{N' \times N'}$, $z_{ij} = e^{-D(i,j)/2\sigma^2}$
$W = \text{diag}\{\sum_i z_{i1}, \sum_i z_{i2}, \ldots, \sum_i z_{iN'}\}$
$
\overline{Z} = W^{-\frac{1}{2}}ZW^{-\frac{1}{2}}$

/* Obtain k eigenvectors with largest eigenvalues */
$V = [v_1, v_2, ..., v_k] = [a_1, a_2, ..., a_{N'}]^T$

/* Normalize each row of the eigenvector matrix */
$\overline{V} = [\overline{a}_1, \overline{a}_2, ..., \overline{a}_{N'}]^T$

/* K-Means clustering */
$d'_{1}, d'_{2}, ..., d'_{N'} = \mathcal{K}([\overline{a}_1, \overline{a}_2, ..., \overline{a}_{N'}])$

/* Distances mapping to original 3D mesh */
$d_1, d_2, ..., d_N = \mathcal{M}([d'_{1}, d'_{2}, ..., d'_{N'}])$

end
4.3. Spectrum-Uncertainty Joint Modeling

In order to fully exploit the potential of large-scale datasets, we hope to simultaneously make use of both the uncertainty and spectral statistics to harvest pseudo labels with higher reliability. Therefore, we investigate the distribution of the joint feature of uncertainty and spectrum distances, and propose a joint double mixture model to fit this bivariate distribution (\(x\) and \(y\) denotes the uncertainty value and the spectrum distance respectively):

\[
P(x,y|\theta_1, \theta_2) = \omega_1 \cdot P(x,y|\theta_1) + \omega_2 \cdot P(x,y|\theta_2)
\]

\[
P(x,y|\theta) = \frac{\Gamma(a + b) x^{a-1} (1 - x)^{b-1}}{\Gamma(a) \Gamma(b)} \cdot \frac{d^c x^{c-1} e^{-dx}}{\Gamma(c)}, \quad (17)
\]

\(\theta = \{a, b, c, d\}\)

\(\theta_1\) and \(\theta_2\) are the parameter sets of the two components. Each parameter set \(\theta\) consists of four independent parameters \((a, b, c\) and \(d\)).

Similar to the other two proposed models, this model is also the weighted sum of a reliable component and an unreliable component, and the weights are denoted by \(\omega_1, \omega_2\) respectively. However, this model takes both dropout-based uncertainty and spectrum distance as input and generates 2-dimension distribution probabilities. With spectrum-uncertainty joint modeling, we take generated labels with higher probabilities to be drawn from the reliable parts as the pseudo labels. We hope that pseudo labels can be harvested with higher precision by jointly fitting the model statistics of two dimensions.

4.4. Double Mixture Model Fitting

In this section, we introduce the math derivations of fitting double mixture models.

Since the three mixture models we proposed have similar forms, we hereby make the following abstractions:

- \(x_i\) is the input (uncertainty, spectrum distance or a pair of the first two) of the \(i\)-th point of the scene
- \(\theta_1, \theta_2\) are the parameter sets of the two components of the mixture model
• \( p(\cdot | \theta_1, \theta_2) \) is the probability density function given the two parameter sets \( \theta_1, \theta_2 \)

Then, our goal is to maximize the log likelihood objective:

\[
L(\theta_1, \theta_2) = \sum_i \log p(x_i | \theta_1, \theta_2)
\]

(18)

Since we use the mixture model, it is difficult to determine which one of the components the input sample belongs to. So similar to [52], we introduce latent variables \( Z = \{z_i\} \). If \( z_i = 1 \), the sample \( x_i \) is drawn from the reliable (certain, localized) part, otherwise if \( z_i = 2 \), the sample \( x_i \) is drawn from the unreliable (uncertain, unlocalized) part. Then the target to optimize is transformed into

\[
L(\theta_1, \theta_2) = \sum_i \log \sum_{z_i \in \{1, 2\}} (p(x_i | z_i, \theta_1, \theta_2) p(z_i | \theta_1, \theta_2))
\]

(19)

The probability of a sample taken from either of the two parts is independent of the sample itself. Therefore, we treat these two probabilities as parameters to optimize and refer to them as \( \omega_1 \) and \( \omega_2 \). More specifically, \( \omega_1 \) denotes the probability that a sample is drawn from the reliable part, and \( \omega_2 \) denotes the probability that a sample is drawn from the unreliable part.

That being said, we rewrite our target to

\[
L(\theta_1, \theta_2, \omega_1, \omega_2) = \sum_i \log \sum_{z_i \in \{1, 2\}} (p(x_i | z_i, \theta_1, \theta_2) \cdot \omega_{z_i})
\]

(20)

and adopt an iterative method to maximize the target by maximizing the difference of \( L(\theta_1, \theta_2, \omega_1, \omega_2) \) in each optimization step:

\[
L(\theta_1, \theta_2, \omega_1, \omega_2) - L(\theta^n_1, \theta^n_2, \omega^n_1, \omega^n_2)
\]

\[
= \sum_i \log \sum_{z_i \in \{1, 2\}} \frac{p(x_i | z_i, \theta_1, \theta_2) \omega_{z_i}}{p(x_i | \theta^n_1, \theta^n_2)}
\]

(21)
In the above equation, \( n \) denotes the number of steps.

Based on the fact that \( \sum z_i p(z_i|x_i, \theta^n_1, \theta^n_2) = 1 \), we use Jensen’s inequality and introduce a new notation \( \mathcal{L}_n \) to get:

\[
\mathcal{L}(\theta_1, \theta_2, \omega_1, \omega_2) - \mathcal{L}(\theta^n_1, \theta^n_2, \omega^n_1, \omega^n_2) \\
\geq \mathcal{L}_n(\theta_1, \theta_2, \omega_1, \omega_2)
\]

\[
= \sum_i \sum_{z_i} p(z_i|x_i, \theta^n_1, \theta^n_2) \log \left( \frac{p(x_i|z_i, \theta_1, \theta_2) \omega_{z_i}}{p(z_i|x_i, \theta^n_1, \theta^n_2) p(x_i|\theta^n_1, \theta^n_2)} \right)
\]

Thus, by increasing \( \mathcal{L}_n(\theta_1, \theta_2, \omega_1, \omega_2) \) the initial target would also increase.

Leaving out irrelevant terms in \( \mathcal{L}_n(\theta_1, \theta_2, \omega_1, \omega_2) \), we have

\[
\mathcal{L}_n'(\theta_1, \theta_2, \omega_1, \omega_2)
\]

\[
= \sum_i \sum_{z_i} p(z_i|x_i, \theta^n_1, \theta^n_2) \log \left( \frac{p(x_i|z_i, \theta_1, \theta_2) \omega_{z_i}}{p(z_i|x_i, \theta^n_1, \theta^n_2) p(x_i|\theta^n_1, \theta^n_2)} \right)
\]

\[
= \sum_i \sum_{z_i} p(z_i|x_i, \theta^n_1, \theta^n_2) \log \left( \frac{p(x_i|z_i, \theta_1, \theta_2) \omega_{z_i}}{p(z_i|x_i, \theta^n_1, \theta^n_2) p(x_i|\theta^n_1, \theta^n_2)} \right)
\]

To simplify the expressions, we introduce a set of new notations:

\[
\mathcal{J}_{n,1}(\theta_1, \theta_2) = \sum_i \sum_{z_i} p(z_i|x_i, \theta^n_1, \theta^n_2) \log \left( \frac{p(x_i|z_i, \theta_1, \theta_2) \omega_{z_i}}{p(z_i|x_i, \theta^n_1, \theta^n_2) p(x_i|\theta^n_1, \theta^n_2)} \right)
\]

\[
\mathcal{J}_{n,2}(\omega_1, \omega_2) = \sum_i \sum_{z_i} p(z_i|x_i, \theta^n_1, \theta^n_2) \log \omega_{z_i}
\]

Then we update the parameters iteratively:

\[
\theta_1^{n+1}, \theta_2^{n+1}, \omega_1^{n+1}, \omega_2^{n+1} = \arg\max_{\theta_1, \theta_2, \omega_1, \omega_2} \{ \mathcal{L}_n(\theta_1, \theta_2, \omega_1, \omega_2) \}
\]

\[
= \arg\max_{\theta_1, \theta_2, \omega_1, \omega_2} \{ \mathcal{L}_n'(\theta_1, \theta_2, \omega_1, \omega_2) \}
\]

\[
= \arg\max_{\theta_1, \theta_2, \omega_1, \omega_2} \{ \mathcal{J}_{n,1}(\theta_1, \theta_2) + \mathcal{J}_{n,2}(\omega_1, \omega_2) \}
\]
Since $\theta_1, \theta_2$ are independent of $\omega_1, \omega_2$, we update the two sets of parameters independently:

$$
\theta_1^{n+1}, \theta_2^{n+1} = \operatorname{argmax}_{\theta_1, \theta_2} \{ \mathcal{J}_{n,1}(\theta_1, \theta_2) \} \\
\omega_1^{n+1}, \omega_2^{n+1} = \operatorname{argmax}_{\omega_1, \omega_2} \{ \mathcal{J}_{n,2}(\omega_1, \omega_2) \} 
$$

(26)

We adopt an EM algorithm to monotonically increase $\mathcal{J}_{n,1}$ and $\mathcal{J}_{n,2}$. In each iteration, we first calculate $\mathcal{J}_{n,1}, \mathcal{J}_{n,2}$ with inputs and current parameters, and then update the parameters by assigning the gradients of $\mathcal{J}_{n,1}, \mathcal{J}_{n,2}$ to zero. After a fixed number of iterations, the fitting process converges.

4.5. Fine-tuning with Pseudo Labels

With the pseudo labels we harvest, we fine-tune the pre-trained Viewpoint Bottleneck framework again. For each point in the scene, we report the category with the maximum logit as the prediction for semantic segmentation.

5. Experiments

5.1. Datasets

**Dataset for SSRL** ScanNetV2 [20] is a large-scale RGB-D video dataset with 3D reconstructions of indoor scenes with over 1500 scans reconstructed from around 2.5 million views. The dataset also provides point-level semantic labels for the scenes, which are defined in a protocol of 20 categories. In our study, we conduct the experiments of self-supervised representation pre-training on the official training split of ScanNetV2 [15] that contains 1201 scans. Our method is evaluated on the official validation split (312 scans) of the dataset. We also provide quantitative results on the held-out test set, which is reported by the official online server.

Stanford Large-Scale 3D Indoor Spaces (S3DIS) [55] dataset contains 3D scans of six indoor areas captured from 3 different buildings. Each area is scanned with RGBD sensors and is represented as point clouds with xyz coordinate and RGB value and annotated with 13 object categories. The evaluation protocol on Area 5 as held-out is adopted. We use 5cm voxel for the experiments.
Semantic3D [56] is a large labelled 3D point cloud dataset of natural scenes, which collected by static terrestrial laser scanners. It contains 8 semantic classes and covers diverse urban scenes. In the commonly used benchmark, there are 9 point clouds in the training set and 6 point clouds in the validation set.

**Dataset for data-efficient fine-tuning** For fair comparison, we use the data-efficient settings of ScannetV2 provided by [20] for fine-tuning. In these settings, only 20/50/100/200 points of a scene in the training splits have labels. Our method leverages the sparse ground-truth annotations to generate reliable pseudo labels. The pseudo labels then serve as supervision during fine-tuning, and other unlabeled points are ignored. For weakly supervised settings of S3DIS and Semantic3D datasets, we evaluate on a subset of points (20/50/100/200 points) uniformly sampled within each points cloud. On these validation sets, the intersection over union (IoU) is taken as the evaluation metrics.

### 5.2. Implementation Details

To prepare the scene meshes for further exploitation, we sample a combination of the following transformations to apply to the point clouds:

- **Random rotation.** We rotate the point cloud along z-axis by a random angle uniformly sampled from \([0, 2\pi]\).

- **Random mirroring.** For each axis, we apply mirroring transformation with a probability of 0.5.

- **Random chromatic jitter.** For each channel, we apply a noise sampled from \(N(0, 255 \times 0.05)\).

Then, the point clouds are downsampled with farthest point sampling (FPS) before being fed to subsequent neural networks. By always selecting the point that is farthest to the already chosen ones, FPS ensures that only redundant points are discarded and the downsampled point cloud is an abstraction of the original point cloud. The target size of downsampled point cloud is 1024 points due to the limitation of GPU memory size used in subsequent neural networks.
We choose Sparse ConvUNet as the backbone of the neural network and implement it with MinkowskiNetwork [57]. Sparse ConvNet has relatively small GPU memory footprints, which enables deeper network and well suits the scenarios where large amounts of points need to be processed simultaneously. The network takes both 3D coordinates and point-wise RGB values as the inputs. The voxel size for Sparse ConvUNet is set to 2.0 cm for the ScannetV2 dataset and 5.0 cm for the S3DIS dataset, since the distances between points in these datasets are within 20.0 cm. For the Semantic3D dataset, we set the voxel size to 50.0 cm due to the larger distances between points (up to 2.0 m). The feature dimension is set to 256/512/1024 to investigate the robustness of Viewpoint Bottleneck.

For SSRL with Viewpoint Bottleneck, experiments are conducted with a batch size of 2 for 20000 iterations on two GeForce RTX 3090 GPUs. An SGD optimizer with a momentum of 0.99 is adopted. The learning rate is decayed polynomially, with the initial value set to 0.1.

For the first and second fine-tuning processes, experiments are conducted with a batch size of 12 for 30000 iterations on two GeForce RTX 3090 GPUs with the same optimizer and scheduler settings.

For uncertainty modeling, we obtain the uncertainty of the labels by running with active dropout layers 10 times with a dropout rate of 0.5. For spectrum modeling, we set the downsample target to 8000, geodesic-to-angular distance proportion to 3:2, and the dimension of the feature space to 50 to optimize the performance. We adopt the implementation in scikit-learn library [58] (i.e. \texttt{sklearn.cluster.KMeans}) for the K-means clustering.

When fitting the mixture models, we adopt the EM algorithm with 50 iterations which is enough for the algorithm to converge.

5.3. Results and Analysis

Results on ScanNet V2 Dataset To show the effectiveness of our proposed method, we evaluate state-of-the-art methods and ours on the ScanNetV2 dataset and report the results in Table. 1. Note that, our methods are compared
### Table 1: 3D Scene Semantic Segmentation Results on ScanNet Dataset

| Setting | Method          | Split |  |  |
|---------|-----------------|-------|---|---|
| **Fully** |                 |       |   |   |
|         | PointNet++ [59]  | -     | 33.9 |   |
|         | PointCNN [60]   | -     | 45.8 |   |
|         | KPConv [61]     | -     | 68.4 |   |
|         | MinkowskiNet [62] | - | 73.6 |   |
|         | BPNet [63]      | -     | 74.9 |   |
| **subs.** |                 |       |   |   |
|         | WyPR [63]       | -     | 31.1 | 24.0 |
|         | MPRM [64]       | -     | 43.2 | 41.1 |
| **10%** | WS [65]         | -     | 52.0 |   |
| **1%**  | WS [65]         | -     | 51.1 |   |
|         | HyraidCR [66]   | -     | 56.9 | 56.8 |
|         | PSU [67]        | -     | 54.7 |   |
| **200 pts (0.1%)** |                 |       |   |   |
|         | SQN [67]        | -     | 59.8 |   |
|         | CSC [20]        | -     | 62.2 | 66.5 |
|         | PointContrast [19] | - | 65.3 |   |
|         | VIB (ours)      | 68.5 | 66.9 |   |
|         | VIBUS (ours)    | 69.6 | 69.1 |   |
| **100 pts (0.05%)** |                 |       |   |   |
|         | SQN [67]        | -     | 57.6 |   |
|         | CSC [20]        | -     | 65.9 | 64.4 |
|         | PointContrast [19] | - | 63.6 |   |
|         | VIB (ours)      | 66.8 | 65.0 |   |
|         | VIBUS (ours)    | 68.9 | 68.4 |   |
| **50 pts (0.03%)** |                 |       |   |   |
|         | SQN [67]        | -     | 54.2 |   |
|         | CSC [20]        | -     | 60.5 | 61.2 |
|         | PointContrast [19] | - | 61.4 |   |
|         | VIB (ours)      | 63.6 | 62.3 |   |
|         | VIBUS (ours)    | 65.6 | 65.1 |   |
| **20 pts (0.01%)** |                 |       |   |   |
|         | SQN [67]        | -     | 48.6 |   |
|         | CSC [20]        | -     | 55.5 | 53.1 |
|         | PointContrast [19] | - | 55.0 |   |
|         | DAT [68]        | -     | 58.6 | -   |
|         | VIB (ours)      | 57.0 | 54.8 |   |
|         | VIBUS (ours)    | 61.0 | 58.6 |   |

### Table 2: 3D Scene Semantic Segmentation Result on ScanNet Validation Set (Trained with Limited Reconstructions, mIOU: %)

| Points  | 20%     | 10%     | 5%      | 1%      |
|---------|---------|---------|---------|---------|
| Scratch | 48.1    | 42.7    | 31.9    | 9.9     |
| CSC [20] | 50.6(+2.5) | 44.9(+2.2) | 36.6(+4.4) | 13.2(+3.3) |
| VIB (ours) | 64.8(+16.7) | 60.5(+17.8) | 47.4(+10.8) | 28.6(+18.7) |
| Setting | Method       | mAP@50% |
|---------|--------------|---------|
| Fully   | MTML[69]     | 40.2    |
| Fully   | 3D-MPA[70]   | 59.1    |
| Fully   | OccuSeg[71]  | 60.7    |
| Fully   | SSTNet[72]   | 64.3    |
| Boxes   | Box2Mask[73] | 59.7    |
| 20 pts  | CSC[20]      | 26.3    |
| 20 pts  | VIB(ours)    | 35.2    |
| 50 pts  | CSC[20]      | 32.6    |
| 50 pts  | VIB(ours)    | 39.7    |
| 100 pts | CSC[20]      | 39.3    |
| 100 pts | VIB(ours)    | 43.3    |
| 200 pts | PointContrast[19] | 44.5 |
| 200 pts | CSC[20]      | 48.9    |
| 200 pts | VIB(ours)    | 45.1    |

with three groups of settings: (1) fully supervised methods trained with 100% point-level annotations (denoted by "Fully") (2) weakly-supervised methods trained with scene/subcloud-level annotations (denoted by "subs.") (3) weakly-supervised methods trained with limited point-level annotations.

In the third setting that has the least supervision with up to 200 annotations (0.1% of the total set of annotations) available per scene, our methods achieve the highest performances on the ScanNet validation set among weakly-supervised methods on the task of semantic segmentation. With 20/50/100 initial annotations, VIBUS achieves large margins over other methods in this setting, especially in the setting of 20 initial annotations (+6.0 mIoU compared with the CSC[20]). With 200 initial annotations, our method also outperforms other methods in this setting but with smaller margins due to the saturation of performances.

We also report the performances of VIBUS and some other baselines on the ScanNet test set in Table. 1, which are provided by the online Scannet Benchmark and ScanNet Data Efficient Benchmark. Note that, the evaluation on the test split is conducted on the test server as the ground truth labels of
the ScanNet test set are not publicly available. Our method (Spec-Unc Field) ranks 1st on the track with 100 initial annotations. We also achieve leading performances on all other tracks with 20/50/200 initial annotations, higher than all the baselines.

We also show the semantic segmentation results under the limited reconstructions setting in Table 2. The performance of VIB achieves significant margins over the baseline CSC[20]. In the Table, we report the instance segmentation results of different methods on the ScanNetV2 validation set. In weakly-supervised settings, our method achieves better performances compared with CSC[20] but still underperforms fully-supervised methods. In Fig. 8, we show the qualitative results of VIBUS on the task of semantic segmentation on the ScanNet dataset in the different settings. In Fig. 10, we show the qualitative results of VIB on the task of instance segmentation with the different levels of supervision. These results prove the novelty of our method under extremely sparse supervision.

Results on Area-5 of S3DIS We evaluate the state-of-the-art methods and our methods on the Area-5 of S3DIS and report the semantic segmentation results in Table. In the four weakly-supervised settings with which we evaluate our methods, we use up to 200 point annotations per scene which is a small portion (up to 0.02%) of the whole set of annotations. In the 100 annotations (0.01%) setting, our method outperforms MT[74] and Xu et al.[75] which are evaluated in the 0.2% setting. In the 200 annotations (0.02%) setting, the performance of our method is the best among all other methods which are evaluated in the 0.2% settings. In general, our method obtains better performance with fewer annotations, proving the effectiveness and the novelty of our method.

We report the performance of our methods on the task of instance segmentation on S3DIS in Table. VIB achieves significant margins over the baseline in all weakly supervised settings. In Fig. 11 and 12, we also show the qualitative results of VIB on the task of semantic segmentation and instance segmentation with different levels of supervision.

Results on Semantic3D (semantic-8) We evaluate our method on the
Table 4: 3D Scene Semantic Segmentation Results on Area-5 of S3DIS

| Setting | Method   | mIoU |
|---------|----------|------|
| Fully   | PointCNN | 57.3 |
| Fully   | KPConv   | 67.1 |
| Fully   | HybridCh | 65.8 |
| Fully   | PSD      | 65.1 |
| Fully   | RandLA-Net | 62.4 |
| 10%     | Xu et al. | 48.0 |
| 10%     | WS3      | 64.0 |
| 1%      | WS3      | 61.8 |
| 1%      | PSD      | 63.6 |
| 0.2%    | MT       | 44.4 |
| 0.2%    | Xu et al. | 44.5 |
| 0.1%    | RandLA-Net | 52.9 |
| 0.03%   | PSD      | 48.2 |
| 0.03%   | HybridCh | 51.5 |
| 20 pts  | VIB(ours) | 29.1 |
| 50 pts  | VIB(ours) | 38.0 |
| 100 pts | VIB(ours) | 46.4 |
| 200 pts | VIB(ours) | 52.0 |

Table 5: 3D Scene Instance Segmentation Result on Area-5 of S3DIS (Trained with Limited Annotation, mAP@50: %)

| Points  | 200   | 100  | 50   | 20   |
|---------|-------|------|------|------|
| Baseline| 29.8  | 15.0 | 8.4  | 5.3  |
| VIB(ours)| 21.4 (+0.6) | 17.0 (+2.0) | 17.6 (+9.2) | 13.9 (+8.6) |
| Setting | Method             | mIoU |
|---------|--------------------|------|
| Fully   | PointNet++[59]     | 63.1 |
| Fully   | Qiu et al.[77]     | 75.4 |
| 10%     | WS3[65]            | 73.3 |
| 1%      | WS3[65]            | 72.6 |
| 0.1%    | SQN[67]            | 72.3 |
| 0.01%   | SQN[67]            | 58.8 |
| 20 pts (0.002%) | baseline          | 14.4 |
| 20 pts (0.002%) | VIB(ours)         | 22.7 |
| 50 pts (0.006%) | baseline          | 26.0 |
| 50 pts (0.006%) | VIB(ours)         | 33.4 |
| 100 pts (0.01%) | baseline          | 32.6 |
| 100 pts (0.01%) | VIB(ours)         | 34.2 |
| 200 pts (0.02%) | baseline          | 36.0 |
| 200 pts (0.02%) | VIB(ours)         | 39.4 |

Table 7: Evaluation of Viewpoint Bottleneck\(^a\) with Different Feature Sizes on ScanNet (mIoU: %)

| Points | 200 | 100 | 50 | 20 |
|--------|-----|-----|----|----|
| Ours (256) | 68.4 | 66.5 | 63.3 | 56.2 |
| Ours (512) | **68.5** | 66.8 | 63.6 | 57.0 |
| Ours (1024) | 68.4 | 66.5 | **63.7** | 56.3 |

\(^a\) Viewpoint Bottleneck is the first stage of VIBUS.

The evaluation of Viewpoint Bottleneck is conducted by fine-tuning the first-stage network with limited ground-truth labels only.

The task of semantic segmentation on Semantic3D (semantic-8) and report the results in Table. \[1\] Directly training with the extremely sparse annotations is considered as the baseline. VIB achieves positive margins over the baselines in different weakly-supervised settings, showing the effectiveness of our method on large-scale outdoor scenes. We show the qualitative results of VIB on the task of semantic segmentation in Fig. \[13\]

5.4. Ablations

To demonstrate the robustness of our method, we conduct several experiments for ablation.
Table 8: Evaluation of VIBUS with Different Modeling Strategies (mIoU: %)

| Points | 200  | 100  | 50   | 20   |
|--------|------|------|------|------|
| VIB (512) + Uncertainty  | 69.6 | 68.6 | 64.7 | 60.7 |
| VIB (512) + Spectrum     | 69.1 | 68.5 | **65.6** | **61.0** |
| VIB (512) + Joint        | 69.4 | **68.9** | 64.6 | 60.0 |

Table 9: Evaluation of Pseudo Labels Harvested by Different Strategies (mIoU: %)

| Points | 200  | 100  | 50   | 20   |
|--------|------|------|------|------|
| Spectrum Modeling         | 51.9 | 51.6 | 46.9 | 39.6 |
| Uncertainty Modeling      | **61.1** | 59.4 | **52.8** | **49.4** |
| Joint Modeling            | 61.0 | 59.2 | 49.8 | 45.2 |

Table 10: Evaluation of Spectral Clustering with Different Target Vertices Number during Downsampling (mIoU: %)

| Downsample Target | 2,000 | 4,000 | 8,000 | 12,000 | 16,000 | 20,000 |
|-------------------|-------|-------|-------|--------|--------|--------|
| 20                | 24.2  | 24.9  | 24.8  | **25.1** | **25.1** | **25.1** |
| 50                | 34.1  | **34.3** | 34.2  | 34.2  | 34.0  | 33.8  |
| 100               | 41.9  | **42.7** | 42.6  | 42.4  | 42.4  | 42.3  |
| 200               | 48.1  | 49.3  | 49.6  | **49.8** | 49.7  | 49.7  |

Table 11: Evaluation of Spectral Clustering with Different Embedding Lengths (mIoU: %)

| Embedding Length | 10   | 30   | 50   | 70   | 90   | 110  | 200  |
|------------------|------|------|------|------|------|------|------|
| 20               | 23.5 | 24.4 | **25.1** | 24.5 | 24.1 | 23.3 | 20.0 |
| 50               | 31.1 | 33.6 | 34.0 | 33.9 | **34.2** | **34.2** | 32.4 |
| 100              | 37.5 | 41.7 | **42.4** | 41.3 | 40.9 | 40.9 | 40.3 |
| 200              | 43.4 | 48.5 | **49.7** | 49.1 | 48.4 | 48.1 | 46.6 |

Table 12: Evaluation of Spectral Clustering with Different Percentage of Geodesic Distance $\delta$ (mIoU: %)

| Proportion of Geodesic Distance $\delta$ | 0.0 | 0.2 | 0.4 | 0.6 | 0.8 | 1.0 |
|-----------------------------------------|----|----|----|----|----|----|
| 20                                      | 13.3 | 24.6 | **25.0** | 24.4 | 24.2 | 23.6 |
| 50                                      | 15.3 | 33.6 | **34.2** | 33.6 | 33.5 | 32.0 |
| 100                                     | 16.0 | 40.4 | **41.7** | **41.7** | 41.5 | 38.5 |
| 200                                     | 16.6 | 46.2 | 48.1 | **48.5** | **48.5** | 44.8 |


**Table 13: Time Consumed in Different Stages**

| Stages          | Time   |
|-----------------|--------|
| pre-training    | 3 days |
| fine-tuning     | 1 day  |
| modeling        | 6 hours|
| inference       | 5 minutes|

**Feature space dimension in for Viewpoint Bottleneck** Table 7 shows the performances of Viewpoint Bottleneck on the validation set with different feature space dimension. We argue that Viewpoint Bottleneck is robust to the feature space dimension since little effect is observed on the final mIoUs. In other experiments, the feature space dimension is set to 512 due to the slight improvements over the others.

**Modeling strategies** We further explore the effectiveness of VIBUS with different strategies for pseudo label harvesting and report the performances on the Scannet semantic segmentation validation set in Table 8 and the semantic segmentation of per-category performance in Table 14. We denote Viewpoint Bottleneck as VIB and denote uncertainty, spectrum, and joint modeling strategies as 'Uncertainty', 'Spectrum', and 'Joint'. In addition, we examine the quality of pseudo labels by directly calculating the mIoU of the pseudo label sets. The results are reported in Table 9.

Results show that the three strategies produce pseudo labels with different mIoU. Specifically, uncertainty modeling produces pseudo labels with the highest mIoU, and spectrum modeling produces pseudo labels with the lowest mIoU. However, fine-tuning with the three sets of pseudo labels achieves comparable final performances. This phenomenon indicates that the three approaches are all capable of filtering reliable pseudo labels.

We also provide visualization results of different modeling strategies in 200-point setting in Fig. 9.

**Downsample ratio in Uncertainty-Spectrum Modeling** Firstly, to evaluate how the downsampling process affects the performance of spectral clustering, we use different numbers of vertices as the target of Quadric Edge Collapse Decimation when before spectral clustering. Table 10 shows the clustering results are robust to the number of target vertices of spectral clustering.
We set this parameter to 8000 in other experiments.

**Embedding length in Uncertainty-Spectrum Modeling** Also, we test different embedding lengths when performing spectral clustering and report the mIoUs of clustering results in Table. With an embedding length of 50, the spectral clustering gives optimal results when provided with 20, 50, and 200 initial annotations. In the setting of 100 initial annotations, the spectral clustering performs the best with an embedding length of 90 or 110. In other experiments, we set the embedding length to 50.

Figure 8: The semantic segmentation qualitative results of VIBUS on the ScanNet validation set under different levels of supervision.
Figure 9: The semantic segmentation qualitative results of Viewpoint Bottleneck, VIB with Spectrum Modeling, VIB with Uncertainty Modeling and VIBUS on the ScanNet validation set. The results are obtained with the setting of 200 initial annotations.

Ground truth | Full supervision | 200 points | 100 points | 50 points | 20 points
---|---|---|---|---|---

Figure 10: The qualitative results of Scannet instance segmentation of VIB in the settings with different initial labels.
Figure 11: The qualitative results of S3DIS semantic segmentation of VIB in the settings with different initial labels.

Figure 12: The qualitative results of S3DIS instance segmentation of VIB in the settings with different initial labels.
Figure 13: The qualitative results of Semantic3D semantic segmentation of VIB in the settings with different initial labels.

**Ratio of geodesic and angular distance** Table. 12 reports the mIoUs of spectral clustering results with different percentage of geodesic distance ($\delta$) in the spectrum distance measure. We observe that using angular distance only ($\delta = 0$) and distance only ($\delta = 1$) both give unsatisfying results. With $\delta = 0.4$ in the settings of 20, 50, 100 initial annotations and with $\delta = 0.6$ in the settings of 100 and 200 initial annotations, spectral clustering has optimal performances. In other experiments, we set $\delta$ to 0.6.

5.5. **Hardware Environment and Time Complexity**

VIBUS is an efficient large-scale 3D scene parsing method. The experiments are conducted on a computing server which have 2 AMD EPYC 7742 CPUs (128 cores in total), 256GB memory and 8 GeForce RTX 3090 GPU Cards. The server runs on Ubuntu 20.04.4 LTS operating system with Linux kernel 5.13.0.

We use 4, 2 and 1 GPU cards in the pretraining, fine-tuning and inference stages respectively. We list the computational overhead of pre-training, modeling, fine-tuning and inference on ScanNetV2 in Table. 13.
6. Limitations and Future Works

Despite the novelty of our proposed methods, there are still some limitations that can be improved in future works.

Firstly, our methods achieve unsatisfying results on small datasets like Semantic3D (which has only 9 scenes for training). Without sufficient scenes, it is hard for the Viewpoint Bottleneck pretraining pipeline to learn how to maintain the information about the input point cloud against the random transformations. In the future, we will seek for a solution that has smaller reliance on the size of the dataset.

Secondly, our proposed methods cannot handle the non-manifold meshes, in which case the calculation of the geodesic distances is inaccurate. To solve this problem, we will try performing mesh completing algorithm, or find a more robust substitute for calculating geodesic distances other than heat diffusion methods.

Table 14: Data-efficient 3D Scene semantic segmentation Result of Different Modelling Approaches on ScanNet Validation Set with 100 Initial Labels (IoU: %)

| setting | category | Wall | Floor | Cabinet | Bed | Chair | S cribs | Desk | Chair | Door | Window | B.S. | Picture | Counter | Desk | Curtain | Fridge | S.C. | Toilet | Sink | Bathtub | O.F. |
|---------|----------|------|-------|---------|-----|-------|--------|------|-------|------|--------|------|---------|---------|------|--------|--------|------|--------|------|---------|------|
| 200 joint | 82.6 | 95.5 | 82.1 | 78.6 | 90.3 | 88.6 | 75.5 | 58.8 | 72.4 | 77.1 | 29.2 | 61.1 | 56.7 | 78.8 | 49.9 | 56.6 | 96.2 | 63.9 | 89.3 | 58.2 |
| uncertainty | 83.8 | 95.7 | 83.2 | 88.1 | 80.3 | 78.9 | 58.2 | 55.1 | 79.1 | 78.4 | 85.6 | 83.1 | 78.7 | 43.5 | 47.6 | 98.2 | 63.9 | 84.5 | 53.5 |
| spectrum | 83.8 | 95.7 | 83.8 | 88.4 | 80.3 | 75.5 | 58.6 | 62.7 | 76.0 | 51.2 | 43.5 | 91.6 | 67.8 | 49.5 | 61.7 | 99.8 | 64.1 | 85.6 | 52.8 |
| 100 joint | 84.6 | 95.8 | 82.9 | 79.5 | 90.7 | 86.2 | 72.2 | 58.7 | 54.9 | 73.6 | 29.9 | 62.9 | 40.9 | 71.4 | 50.4 | 47.7 | 96.1 | 64.4 | 86.5 | 54.5 |
| uncertainty | 82.2 | 95.9 | 81.8 | 79.9 | 89.3 | 80.1 | 72.7 | 57.4 | 54.2 | 73.7 | 28.5 | 83.7 | 42.0 | 71.1 | 48.3 | 63.2 | 98.4 | 58.0 | 82.0 | 48.2 |
| spectrum | 82.7 | 95.8 | 82.9 | 86.2 | 84.0 | 70.8 | 55.6 | 72.8 | 71.7 | 27.6 | 46.4 | 73.7 | 47.4 | 67.6 | 89.3 | 56.5 | 84.6 | 47.7 |
| 50 joint | 79.0 | 95.5 | 74.9 | 77.6 | 88.9 | 83.6 | 65.5 | 52.7 | 54.4 | 74.4 | 29.7 | 80.2 | 57.6 | 74.4 | 41.4 | 45.3 | 96.6 | 54.8 | 82.0 | 44.3 |
| uncertainty | 80.2 | 95.7 | 75.7 | 77.8 | 86.7 | 81.4 | 70.1 | 52.8 | 56.9 | 75.0 | 28.8 | 28.0 | 71.6 | 78.4 | 46.2 | 66.3 | 89.8 | 57.7 | 83.2 | 44.8 |
| spectrum | 80.6 | 95.6 | 75.4 | 76.3 | 87.8 | 80.5 | 66.2 | 52.4 | 48.7 | 71.5 | 28.8 | 63.0 | 56.2 | 89.8 | 52.5 | 86.7 | 97.2 | 87.6 | 84.1 | 46.5 |
| 20 joint | 76.8 | 95.4 | 71.1 | 76.7 | 86.7 | 79.6 | 69.9 | 62.5 | 43.4 | 69.5 | 65.0 | 30.5 | 87.5 | 63.5 | 56.2 | 50.6 | 87.4 | 71.4 | 79.8 | 49.9 |
| uncertainty | 77.1 | 95.4 | 70.8 | 76.9 | 86.9 | 80.2 | 67.3 | 42.2 | 43.7 | 69.1 | 61.0 | 51.2 | 81.2 | 64.2 | 43.0 | 40.5 | 69.5 | 54.8 | 80.1 | 45.6 |
| spectrum | 77.4 | 95.4 | 72.1 | 76.0 | 85.8 | 78.7 | 46.1 | 41.7 | 44.4 | 70.9 | 51.4 | 51.5 | 90.5 | 63.6 | 55.6 | 84.4 | 57.0 | 79.8 | 45.4 |

* The B.S. is the Bookshelf, the S.C. is the shower curtain and the O.F. is the other furniture.

7. Conclusion

In this work, we address the problem of data-efficient 3D scene parsing and propose a principled method named VIBUS with two technical modules: 1) Viewpoint Bottleneck, a self-representation learning pipeline that has simple network structures and avoids selecting sample pairs, and 2) Uncertainty-Spectrum modeling, an effective approach to harvest reliable pseudo labels by
filtering a set of generated label. With simple implementation, our method achieves large margins over existing methods on the ScanNetV2 dataset and wins the leading place on the online server.

References

[1] D. Seichter, M. Köhler, B. Lewandowski, T. Wengefeld, H.-M. Gross, Efficient rgb-d semantic segmentation for indoor scene analysis, in: 2021 IEEE International Conference on Robotics and Automation (ICRA), 2021, pp. 13525–13531. doi:10.1109/ICRA48506.2021.9561675.

[2] D. S. Chaplot, D. P. Gandhi, A. Gupta, R. R. Salakhutdinov, Object goal navigation using goal-oriented semantic exploration, Advances in Neural Information Processing Systems 33 (2020) 4247–4258.

[3] D. Acharya, K. Khoshelham, S. Winter, Bim-posenet: Indoor camera localisation using a 3d indoor model and deep learning from synthetic images, ISPRS Journal of Photogrammetry and Remote Sensing 150 (2019) 245–258.

[4] Q. Chen, J. Chen, W. Huang, Method for generation of indoor gis models based on bim models to support adjacent analysis of indoor spaces, ISPRS International Journal of Geo-Information 9 (9) (2020) 508.

[5] Y. Wang, Z. Zhang, R. Zhong, L. Sun, S. Leng, Q. Wang, Densely connected graph convolutional network for joint semantic and instance segmentation of indoor point clouds, ISPRS Journal of Photogrammetry and Remote Sensing 182 (2021) 67–77. doi:https://doi.org/10.1016/j.isprsjprs.2021.10.003.

[6] H. Lin, S. Wu, Y. Chen, W. Li, Z. Luo, Y. Guo, C. Wang, J. Li, Semantic segmentation of 3d indoor lidar point clouds
through feature pyramid architecture search, ISPRS Journal of Photogrammetry and Remote Sensing 177 (2021) 279–290.

doi:https://doi.org/10.1016/j.isprsjprs.2021.05.009
URL https://www.sciencedirect.com/science/article/pii/S0924271621001349

[7] J. Du, G. Cai, Z. Wang, S. Huang, J. Su, J. Marcato Junior, J. Smit, J. Li, Resdlps-net: Joint residual-dense optimization for large-scale point cloud semantic segmentation, ISPRS Journal of Photogrammetry and Remote Sensing 182 (2021) 37–51.
doi:https://doi.org/10.1016/j.isprsjprs.2021.09.024
URL https://www.sciencedirect.com/science/article/pii/S0924271621002628

[8] L. Yi, V. G. Kim, D. Ceylan, I.-C. Shen, M. Yan, H. Su, C. Lu, Q. Huang, A. Sheffer, L. Guibas, A scalable active framework for region annotation in 3d shape collections, ACM Transactions on Graphics (ToG) 35 (6) (2016) 1–12.

[9] Y. Lin, G. Vosselman, M. Y. Yang, Weakly supervised semantic segmentation of airborne laser scanning point clouds, ISPRS Journal of Photogrammetry and Remote Sensing 187 (2022) 79–100.
doi:https://doi.org/10.1016/j.isprsjprs.2022.03.001
URL https://www.sciencedirect.com/science/article/pii/S0924271622000661

[10] D. Zingsheim, P. Stotko, S. Krumpen, M. Weinmann, R. Klein, Collaborative vr-based 3d labeling of live-captured scenes by remote users, IEEE Computer Graphics and Applications 41 (4) (2021) 90–98.

[11] P. Z. Ramirez, C. Paternesi, L. De Luigi, Lella, D. De Gregorio, L. Di Stefano, Shooting labels: 3d semantic labeling by virtual reality, in: 2020 IEEE International Conference on Artificial Intelligence and Virtual Reality (AIVR), IEEE, 2020, pp. 99–106.
[12] O. Miksik, V. Vineet, M. Lidegaard, R. Prasaath, M. Nießner, S. Golodetz, S. L. Hicks, P. Pérez, S. Izadi, P. H. Torr, The semantic paintbrush: Interactive 3d mapping and recognition in large outdoor spaces, in: Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, 2015, pp. 3317–3326.

[13] M. Shreve, B. Price, L. Nelson, R. Bala, J. Sun, S. Kumar, Warhol: Wearable holographic object labeler, Electronic Imaging 32 (2020) 1–10.

[14] J. Valentin, V. Vineet, M.-M. Cheng, D. Kim, J. Shotton, P. Kohli, M. Nießner, A. Criminisi, S. Izadi, P. Torr, Semanticpaint: Interactive 3d labeling and learning at your fingertips, ACM Transactions on Graphics (TOG) 34.

[15] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, M. Nießner, Scannet: Richly-annotated 3d reconstructions of indoor scenes, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 5828–5839.

[16] G. E. Hinton, R. R. Salakhutdinov, Reducing the dimensionality of data with neural networks, science 313 (5786) (2006) 504–507.

[17] C. Doersch, A. Gupta, A. A. Efros, Unsupervised visual representation learning by context prediction, in: Proceedings of the IEEE international conference on computer vision, 2015, pp. 1422–1430.

[18] K. He, H. Fan, Y. Wu, S. Xie, R. Girshick, Momentum contrast for unsupervised visual representation learning, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, pp. 9729–9738.

[19] S. Xie, J. Gu, D. Guo, C. R. Qi, L. Guibas, O. Litany, Pointcontrast: Unsupervised pre-training for 3d point cloud understanding, in: European conference on computer vision, Springer, 2020, pp. 574–591.
[20] J. Hou, B. Graham, M. Nießner, S. Xie, Exploring data-efficient 3d scene understanding with contrastive scene contexts, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 15587–15597.

[21] Z. Li, D. Hoiem, Improving confidence estimates for unfamiliar examples, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 2686–2695.

[22] Y. Dong, F. Liao, T. Pang, H. Su, J. Zhu, X. Hu, J. Li, Boosting adversarial attacks with momentum, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 9185–9193.

[23] J. Zbontar, L. Jing, I. Misra, Y. LeCun, S. Deny, Barlow twins: Self-supervised learning via redundancy reduction, in: International Conference on Machine Learning, PMLR, 2021, pp. 12310–12320.

[24] A. Bardes, J. Ponce, Y. LeCun, Vicreg: Variance-invariance-covariance regularization for self-supervised learning, arXiv preprint arXiv:2105.04906.

[25] Y. Li, R. Pogodin, D. J. Sutherland, A. Gretton, Self-supervised learning with kernel dependence maximization, Advances in Neural Information Processing Systems 34.

[26] Y. Gal, Z. Ghahramani, Dropout as a bayesian approximation: Representing model uncertainty in deep learning, in: international conference on machine learning, PMLR, 2016, pp. 1050–1059.

[27] P. Cignoni, A. Muntoni, G. Ranzuglia, M. Callieri, Meshlab. doi:10.5281/zenodo.5114037

[28] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, A. A. Efros, Context encoders: Feature learning by inpainting, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2536–2544.
[29] R. Zhang, P. Isola, A. A. Efros, Split-brain autoencoders: Unsupervised learning by cross-channel prediction, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1058–1067.

[30] N. Komodakis, S. Gidaris, Unsupervised representation learning by predicting image rotations, in: International Conference on Learning Representations (ICLR), 2018.

[31] S. Chopra, R. Hadsell, Y. LeCun, Learning a similarity metric discriminatively, with application to face verification, in: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05), Vol. 1, IEEE, 2005, pp. 539–546.

[32] X. Wang, R. Zhang, C. Shen, T. Kong, L. Li, Dense contrastive learning for self-supervised visual pre-training, in: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021, pp. 3023–3032. doi:10.1109/CVPR46437.2021.00304.

[33] I. Alonso, A. Sabater, D. Ferstl, L. Montesano, A. C. Murillo, Semi-supervised semantic segmentation with pixel-level contrastive learning from a class-wise memory bank, in: 2021 IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 8199–8208. doi:10.1109/ICCV48922.2021.00811.

[34] J.-B. Grill, F. Strub, F. Alteché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Avila Pires, Z. Guo, M. Gheshlaghi Azar, et al., Bootstrap your own latent—a new approach to self-supervised learning, Advances in Neural Information Processing Systems 33 (2020) 21271–21284.

[35] X. Chen, K. He, Exploring simple siamese representation learning, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 15750–15758.

[36] A. Van den Oord, Y. Li, O. Vinyals, Representation learning with contrastive predictive coding, arXiv e-prints (2018) arXiv–1807.
[37] T. Chen, S. Kornblith, M. Norouzi, G. Hinton, A simple framework for contrastive learning of visual representations, in: International conference on machine learning, PMLR, 2020, pp. 1597–1607.

[38] Z. Zhang, R. Girdhar, A. Joulin, I. Misra, Self-supervised pretraining of 3d features on any point-cloud, in: 2021 IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 10232–10243. doi: 10.1109/ICCV48922.2021.01009.

[39] Y. Liu, L. Yi, S. Zhang, Q. Fan, T. Funkhouser, H. Dong, P4contrast: Contrastive learning with pairs of point-pixel pairs for rgb-d scene understanding, arXiv preprint arXiv:2012.13089.

[40] Y. Liu, Q. Fan, S. Zhang, H. Dong, T. Funkhouser, L. Yi, Contrastive multimodal fusion with tupleinfonce, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 754–763.

[41] S. Huang, Y. Xie, S.-C. Zhu, Y. Zhu, Spatio-temporal self-supervised representation learning for 3d point clouds, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 6535–6545.

[42] H. Zhao, J. Shi, X. Qi, X. Wang, J. Jia, Pyramid scene parsing network, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 2881–2890.

[43] R. Jena, S. P. Awate, A bayesian neural net to segment images with uncertainty estimates and good calibration, in: International Conference on Information Processing in Medical Imaging, Springer, 2019, pp. 3–15.

[44] B. Lakshminarayan, A. Pritzel, C. Blundell, Simple and scalable predictive uncertainty estimation using deep ensembles, Advances in neural information processing systems 30.

[45] L. Rong, Z. Hao, Segmentation of 3d meshes through spectral clustering, in: Computer Graphics & Applications, Pacific Conference, 2004.
[46] H. Shinnou, M. Sasaki, Spectral clustering for a large data set by reducing the similarity matrix size, in: Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC’08), 2008.

[47] X. Chen, D. Cai, Large scale spectral clustering with landmark-based representation, in: Twenty-fifth AAAI conference on artificial intelligence, 2011.

[48] M. Garland, P. S. Heckbert, Surface simplification using quadric error metrics, in: Proceedings of the 24th annual conference on Computer graphics and interactive techniques, 1997, pp. 209–216.

[49] N. Tishby, N. Zaslavsky, Deep learning and the information bottleneck principle, in: Information Theory Workshop, 2015.

[50] N. Tishby, F. C. Pereira, W. Bialek, The information bottleneck method, arXiv.

[51] T. T. Cai, T. Liang, H. H. Zhou, Law of log determinant of sample covariance matrix and optimal estimation of differential entropy for high-dimensional gaussian distributions, Journal of Multivariate Analysis 137 (2015) 161–172.

[52] H. Zhao, M. Lu, A. Yao, Y. Guo, Y. Chen, L. Zhang, Pointly-supervised scene parsing with uncertainty mixture, Computer Vision and Image Understanding 200 (2020) 103040.

[53] K. Crane, C. Weischedel, M. Wardetzky, Geodesics in heat: A new approach to computing distance based on heat flow, ACM Transactions on Graphics (TOG) 32 (5) (2013) 1–11.

[54] A. Muntoni, P. Cignoni, PyMeshLab (Jan. 2021). doi:10.5281/zenodo.4438750.

[55] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, S. Savarese, 3d semantic parsing of large-scale indoor spaces, in: 2016 IEEE
Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 1534–1543. doi:10.1109/CVPR.2016.170

[56] T. Hackel, N. Savinov, L. Ladicky, J. D. Wegner, K. Schindler, M. Pollefeys, SEMANTIC3D.NET: A new large-scale point cloud classification benchmark, in: ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. IV-1-W1, 2017, pp. 91–98.

[57] C. Choy, J. Gwak, S. Savarese, 4d spatio-temporal convnets: Minkowski convolutional neural networks, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 3075–3084.

[58] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, G. Varoquaux, API design for machine learning software: experiences from the scikit-learn project, in: ECML PKDD Workshop: Languages for Data Mining and Machine Learning, 2013, pp. 108–122.

[59] C. R. Qi, L. Yi, H. Su, L. J. Guibas, Pointnet++: Deep hierarchical feature learning on point sets in a metric space, in: I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems, Vol. 30, Curran Associates, Inc., 2017.

URL https://proceedings.neurips.cc/paper/2017/file/d8bf84be3800d12f74d8b05e9b89836f-Paper.pdf

[60] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, B. Chen, Pointcnn: Convolution on x-transformed points, in: NeurIPS, 2018.

[61] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, L. Guibas, Kpconv: Flexible and deformable convolution for point clouds, in: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 6410–6419. doi:10.1109/ICCV.2019.00651
[62] W. Hu, H. Zhao, L. Jiang, J. Jia, T.-T. Wong, Bidirectional projection network for cross dimensional scene understanding, in: CVPR, 2021.

[63] Z. Ren, I. Misra, A. G. Schwing, R. Girdhar, 3d spatial recognition without spatially labeled 3d, in: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021, pp. 13199–13208. doi:10.1109/CVPR46437.2021.01300

[64] J. Wei, G. Lin, K.-H. Yap, T.-Y. Hung, L. Xie, Multi-path region mining for weakly supervised 3d semantic segmentation on point clouds, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 4383–4392. doi:10.1109/CVPR42600.2020.00444

[65] Y. Zhang, Z. Li, Y. Xie, Y. Qu, C. Li, T. Mei, Weakly supervised semantic segmentation for large-scale point cloud, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 35, 2021, pp. 3421–3429.

[66] M. Li, Y. Xie, Y. Shen, B. Ke, R. Qiao, B. Ren, S. Lin, L. Ma, Hybridcr: Weakly-supervised 3d point cloud semantic segmentation via hybrid contrastive regularization, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022, pp. 14930–14939.

[67] Y. Zhang, Y. Qu, Y. Xie, Z. Li, S. Zheng, C. Li, Perturbed self-distillation: Weakly supervised large-scale point cloud semantic segmentation, in: 2021 IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 15500–15508. doi:10.1109/ICCV48922.2021.01523

[68] Z. Wu, Y. Wu, G. Lin, J. Cai, C. Qian, Dual adaptive transformations for weakly supervised point cloud segmentation (2022). arXiv:2207.09084

[69] J. Lahoud, B. Ghanem, M. R. Oswald, M. Pollefeys, 3d instance segmentation via multi-task metric learning, in: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 9255–9265. doi:10.1109/ICCV.2019.00935
[70] F. Engelmann, M. Bokeloh, A. Fathi, B. Leibe, M. Nießner, 3d-mpa: Multi-proposal aggregation for 3d semantic instance segmentation, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 9028–9037. doi:10.1109/CVPR42600.2020.00905

[71] L. Han, T. Zheng, L. Xu, L. Fang, Occuseg: Occupancy-aware 3d instance segmentation, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 2937–2946. doi:10.1109/CVPR42600.2020.00301

[72] Z. Liang, Z. Li, S. Xu, M. Tan, K. Jia, Instance segmentation in 3d scenes using semantic superpoint tree networks, in: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 2783–2792.

[73] J. Chibane, F. Engelmann, T. A. Tran, G. Pons-Moll, Box2mask: Weakly supervised 3d semantic instance segmentation using bounding boxes, in: European Conference on Computer Vision (ECCV), Springer, 2022.

[74] A. Tarvainen, H. Valpola, Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results, in: NIPS, 2017.

[75] X. Xu, G. H. Lee, Weakly supervised semantic point cloud segmentation: Towards 10x fewer labels, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

[76] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, A. Markham, Randla-net: Efficient semantic segmentation of large-scale point clouds, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 11105–11114. doi:10.1109/CVPR42600.2020.01112

[77] S. Qiu, S. Anwar, N. Barnes, Semantic segmentation for real point cloud scenes via bilateral augmentation and adaptive fusion, in: 2021 IEEE/CVF
Conference on Computer Vision and Pattern Recognition (CVPR), 2021, pp. 1757–1767. doi:10.1109/CVPR46437.2021.00180.