2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) | 978-1-7281-6212-6/20/$31.00 ©2020 IEEE | DOI: 10.1109/IROS45743.2020.9341060

Semantic Graph Based Place Recognition for 3D Point Clouds

Xin Kong[1], Xuemeng Yang[1], Guangyao Zhai[1], Xiangrui Zhao[1],
Xianfang Zeng[1], Mengmeng Wang[1], Yong Liu[1]*, Wanlong Li[2] and Feng Wen[2]

***Abstract*— Due to the difficulty in generating the effective descriptors which are robust to occlusion and viewpoint changes, place recognition for 3D point cloud remains an open issue. Unlike most of the existing methods that focus on extracting local, global, and statistical features of raw point clouds, our method aims at the semantic level that can be superior in terms of robustness to environmental changes. Inspired by the perspective of humans, who recognize scenes through identifying semantic objects and capturing their relations, this paper presents a novel semantic graph based approach for place recognition. First, we propose a novel semantic graph representation for the point cloud scenes by reserving the semantic and topological information of the raw point cloud. Thus, place recognition is modeled as a graph matching problem. Then we design a fast and effective graph similarity network to compute the similarity. Exhaustive evaluations on the KITTI dataset show that our approach is robust to the occlusion as well as viewpoint changes and outperforms the state-of-the-art methods with a large margin. Our code is available at: `https://github.com/kxhit/SG_PR`.**

I. Introduction

In the past few decades, Simultaneous Localization and Mapping (SLAM) has developed rapidly, which plays a critical role in robotic applications and autonomous driving. Loop closure detection is an important issue in SLAM, which refers to the ability of robots or moving vehicles to recognize whether a place has been reached before. It is the most effective way to eliminate the cumulative odometry drift error, helping build a more precise global map and achieve more accurate localization. Current strategies for place recognition are primarily based on descriptors generation and feature distance measurement.

Research on vision-based place recognition has been investigated for a long time and many successful approaches have been proposed [1]–[3]. Most of the image-based methods extract feature descriptors and then encode them with methods such as bag-of-words (BoW) [1], [4], VLAD [5] and Fisher Vector (FV) [6], [7]. The relevant scenes are retrieved by comparing the global descriptors and measuring the similarity among them. However, due to the interference of external conditions such as weather, seasons, illumination, and viewpoint changes, image-based methods are probably failed to retrieve the correct match [8].

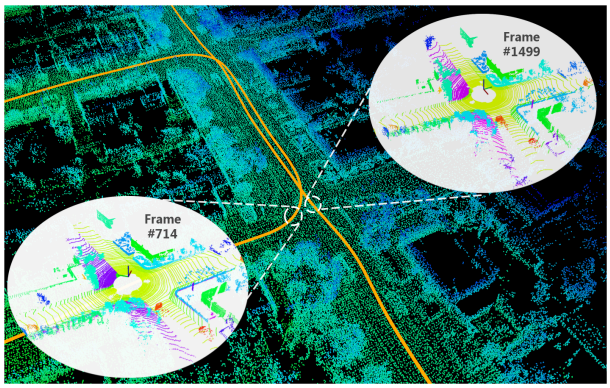


Fig. 1: An illustration of place recognition. This is a reverse loop closure in sequence 08 of KITTI detected by our proposed approach. Note that the heading direction of frame 714 and 1499 are almost exactly the opposite, which brings a challenge to existing methods. (Best viewed with zoom-in.)

LiDAR-based methods are recently gained widespread attention, as they are more robust to seasons and illumination variations. Most LiDAR-based algorithms [9]–[12] operate directly on raw point cloud data and generate local or global descriptors by neural networks or handcrafted design. Such methods obtain low-level features like local structures and distributing characteristics of points, which are sensitive to occlusion and rotation. A few segment based approaches [13]–[15] recognize places by matching segments that belong to partial or full objects, which can better represent dynamic situations. These methods are more related to the way humans perceive their surroundings. However, it's hard to obtain the accurate and stable feature of segments and they ignore relations among segments, which is crucial to the scene expression.

Humans perceive the environment by distinguishing scenes through semantic objects and their topological relations. Inspired by this, we present a new approach that converts the raw point cloud data to a novel graph representation by aggregating the semantic information. Such graph representation retains critical information and considers the topological relations, making the expression of the point cloud data more efficient and comprehensible. Moreover, we apply a learning-based graph similarity computation strategy to solve the retrieve task instead of simply calculating the Euclidean distances of feature vectors. To the best of our knowledge, we are the first to use semantic graph represen-

[1]The authors are with the Institute of Cyber-Systems and Control, Zhejiang University, Hangzhou, 310027, China. (Yong Liu* is the corresponding author, email: yongliu@iipc.zju.edu.cn)

[2]The authors are with Huawei Noah's Ark Lab, Beijing, China.

This work is supported by the National Natural Science Foundation of China under Grant 61836015. We thank Huan Yin for fruitful discussion.

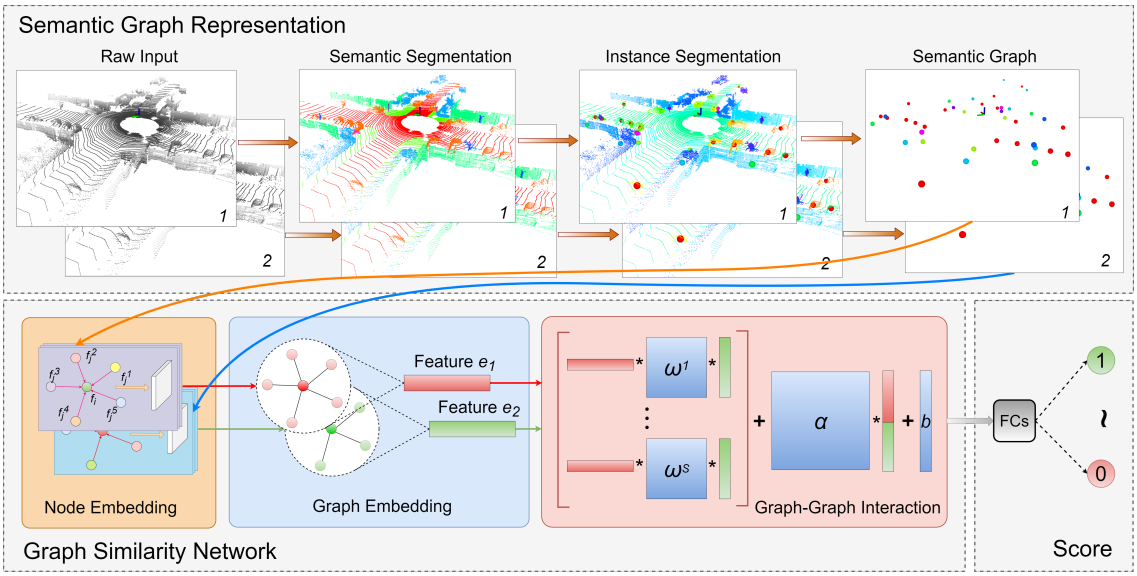


Fig. 2: The architecture of our approach. The whole framework consists of semantic graph representation and graph similarity network. We first segment the point clouds semantically to obtain instances on raw point clouds. In addition, we aggregate the semantic and topological information to acquire nodes and form the semantic graph. By feeding pairs of graphs into graph similarity network composed of node embedding, graph embedding, and graph-graph interaction, we acquire a similarity score in the range $[0, 1]$. FCs denotes a set of fully connected layers.

tation and graph matching for place recognition in 3D point clouds. A demonstration of our results is displayed in Fig. 1. Our contributions can be summarized as follows:

(1) Towards humanoid perception, we present a novel semantic graph representation for 3D point cloud scenes, which captures semantic information and models topological relations between semantic objects.

(2) We propose an effective and efficient network to estimate the graph matching similarity among point cloud scenes which can be used in loop closure detection.

(3) Experiments on the KITTI odometry dataset [16] show that our approach achieves state-of-the-art performance, especially for reverse loop closure detection and the robustness to occlusion as well as viewpoint changes.

II. Related Work

The loop closure detection methods based on 3D point cloud can be divided into the following categories: local descriptor based methods, global descriptor based methods, and segments based methods.

Local descriptor based methods: Spin image [17] first generates a cylindrical coordinate system around each key point, separates the nearby points into bins, and then encodes a pattern of surrounding bins into a histogram. Bosse and Zlot [18] query a constant number of nearest neighbor votes for each keypoint from the database of local 3D Gestalt descriptors and places with a sufficient number of votes are determined as possible location matches. Whereas such local keypoint features often lack descriptive power to distinguish similar local structures and are not always reliable for matching.

Global descriptor based methods: ESF [19] presents a global shape descriptor using a concatenation of histograms describing distance, angle, and area distributions on the surface of the partial point cloud. Without extracting the normal vectors of each point, the lack of spatial information in these descriptors makes it hard to capture intricate details in different clouds. M2DP [9] projects a 3D point cloud onto multiple 2D planes and generates density signatures. The left and right singular vectors of these signatures are then used as descriptors for the 3D scene. However, it relies on the distribution of all points and the performance is not satisfied when there is a partial loss of points. LiDAR Iris [20] extracts corresponding binary signature images from point clouds and measure similarities by calculating hamming-distance. PointNetVLAD [10] combines PointNet [21] and NetVLAD [22], which is the first point cloud network to directly handle the point cloud scenes in 3D space. But the local feature extraction and the spatial distribution of local features are not fully considered. SeqLPD [12] and LPD-Net [11] extract features in both feature space and Cartesian space, fuse the neighborhood features of each point, and use NetVLAD to generate the global descriptors. The above methods process a large number of raw points and achieve unsatisfactory performance when the point cloud scenes rotate. Scan Context [23] reserves maximum heights and maps 3D point clouds to 2D planes by histogram statistics. To achieve rotation invariance, it calculates all possible column-shifted scan contexts line-by-line to find the minimum distance, requiring longer search time than others.

Segments based methods: SegMatch [13] and SegMap [14] present a high-level perception which segments

point clouds into a set of distinct and discriminative elements at object-level. They use a 3D CNN to encode segment features and identify candidate correspondences by using k nearest neighbors (kNN) in feature space. This approach is a successful attempt towards humanoid perception. Nonetheless, it needs a dense local map, and relations among objects are not taken into consideration.

To address the above problems, we create a novel graph representation at a semantic level, making it more concise and effective. Then we apply a graph similarity network instead of Euclidean distance to measure similarities of scenes for better estimation.

III. Methodology

In this section, we present our semantic graph based place recognition approach, which consists of semantic graph representation and learning-based graph similarity computation, as shown in Fig. 2. Our key insight is to recognize the scenes from the human perspective, describe it at the semantic level, and focus on encoding relations among semantic objects. People usually recognize scenes by identifying some semantic objects and observing their relative positions. For this reason, we utilize semantic segmentation on raw point clouds to obtain instances and further collect semantic and topological information together to acquire nodes forming the semantic graph. After this, the raw point cloud scene is transformed into a topological semantic graph and the place recognition task is now converted to a graph matching problem. Moreover, a learning-based graph similarity computation is applied to obtain the similarity scores between pairs of scenes.

To the best of our knowledge, this is the first attempt to use topological semantic graph representation and graph matching in loop closure detection.

A. Semantic Graph Representation

Semantic Segmentation for Point Cloud: Some semantic segmentation methods for point cloud have been proposed recently [24]–[27]. RangeNet++ [27] is trained on SemanticKITTI [28] dataset, which annotates the semantic categories of each 3D point on the KITTI [16] odometry dataset, including a total of 19 classes. In the experimental part, we use predictions of RangeNet++ (which can be flexibly replaced by other methods) and annotations of SemanticKITTI as semantic information respectively. We merge dynamic classes to their corresponding static ones and ignore classes like person, because they are either irrelevant or few in number. The merged 12 categories are shown in Fig. 3. Then, we set different clustering radii according to semantic categories and obtain semantic instances by Euclidean clustering. Specifically, for one single frame of point cloud $P = \{p_1, \ldots, p_M | p_i \in \mathbb{R}^3\}$, we acquire the semantic label of each point p_i, and cluster points with the same semantic label into a set of clusters $I = \{I_1, \ldots, I_N\}$, $I_i = \{p_1, \ldots, p | p_i \in \mathbb{R}^3\} \subset I$ which indicates different instances, and their corresponding semantic labels are $L = \{l_1, \ldots, l_N | l_i \in \mathbb{R}^1\}$.

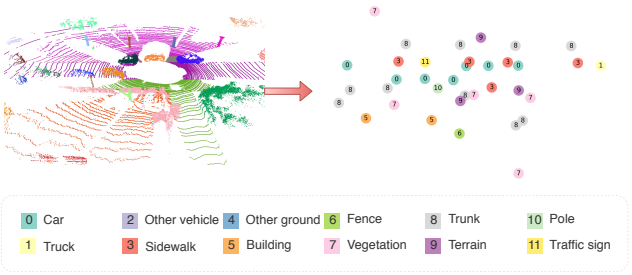


Fig. 3: An illustration of semantic graph representation for point clouds. Each node denotes an instance in the scene, reserving its semantic and topological information.

Semantic Graph Construction: A 64-ring LiDAR usually captures more than 100k points per frame, which is huge and redundant. To reduce the data, most of the existing methods downsample the points randomly [9], [10] or project them onto a 2D plane [23], [29]. Distinctively, we construct topological semantic graph representation to reserve the key information by retaining the semantic information and topological relations of the semantic instances, which is more concise and meaningful.

As shown in Fig. 3, for each instance I_i, we represent it with a node V_i, preserving its semantic category l_i and the centroid $(\bar{x}_i, \bar{y}_i, \bar{z}_i)$ of the set of points $(x_j, y_j, z_j) \in I_i$, composing the node feature $f_i \in \mathbb{R}^4$. In the node embedding part, the semantic feature is one-hot encoded (e.g. all the nodes with car type share the same one-hot encoding vector). Thus, these nodes together form a graph that can represent a point cloud scene. The LiDAR-based loop closure detection is determined by comparing the similarity of two scenes, which has now been turned into a similarity measurement problem for two topological semantic graphs.

B. Graph Similarity Network

Graphs have a wide range of applications and there are different similarity metrics, such as Graph Edit Distance (GED) [30], Maximum Common Subgraph (MCS) [31]. However, computing these metrics between two graphs is NP-complete [31], [32] and it is hard to compute the exact distance within reasonable time [33]. Beyond this, some challenges need to be tackled as well when implementing graph matching in loop closure detection. The algorithm needs to be representation invariant as the computed similarity score should be permutation invariant to the order of nodes. Besides, it should be rotation invariant because reverse loop closure is common in real-world applications. The computational efficiency and generalization ability are crucial as well. Based on the above considerations, we propose the graph similarity network to perform graph matching for place recognition inspired by SimGNN [34]. Our network structure is shown in Fig. 2.

Node Embedding: Graph Convolutional Network (GCN) [35] is the most popular method for aggregating node features. However, the adjacent matrix should be defined and fixed in advance. Given that our nodes can be seen

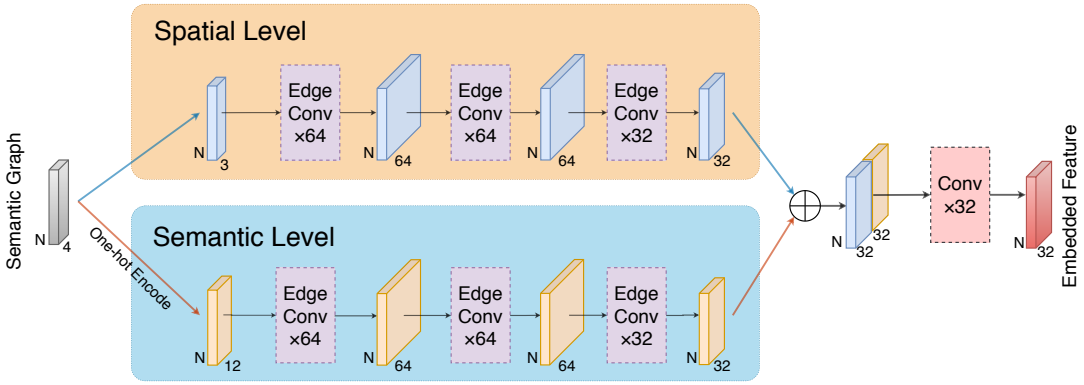


Fig. 4: Detailed architecture of node embedding. We obtain node features on spacial and semantic level respectively, and concatenate them together to get the node embedding.

as superpoints and Dynamic Graph CNN (DGCNN) [36] is effective in point cloud feature learning, we adopt EdgeConv introduced in DGCNN to capture local geometric information while guaranteeing permutation invariance. Our node embedding module applies a dynamically updated graph instead of a fixed graph to group nodes semantically. The detailed architecture is shown in Fig. 4.

In the EdgeConv layer, we find a set of k nearest neighbors (kNN) for each node V_i in feature and Euclidean space, and aggregate features within each set. Each node feature f_i is initialized with centroid information and one-hot encoding based on the semantic label l_i. Each edge represents the feature relation of f_i and its k-nearest neighbors f_j^m, $m = 1, 2, ..., k$ in feature space and edge function is defined as

$$h_\Theta\left(f_i, f_j\right) = \bar{h}_\Theta\left(f_i, f_i - f_j^m\right), \tag{1}$$

where Θ is a series of learnable parameters. This operation combines the global information captured by f_i and the local relations captured by $f_i - f_j^m$. As independent convolution is more efficient for multimodal features [37]–[39], we perform feature aggregation separately on the spatial and semantic level, and concatenate the output features together as the final embedding u_i of each node.

Graph Embedding: A weighted or unweighted average of node embedding is usually used for generating a graph embedding. Inspired by SimGNN [34], we would like to estimate a learnable weight matrix associated with each node by an attention module. We let the neural network learn which node should receive higher attention and be more representative of the overall graph.

For each node V_i, $u_i \in \mathbb{R}^D$ indicates the node embedding of V_i, where D is the dimension of u_i. We initialize the global graph context $c \in \mathbb{R}^D$ as a simple average of each node embedding and followed by an activation function, where we use a $tanh()$. So the global graph context c is

$$c = tanh\left(\left(\frac{1}{N}\sum_{i=1}^{N} u_i\right) W\right), \tag{2}$$

where $W \in \mathbb{R}^{D\times D}$ is a learnable weight matrix, N is the number of nodes in a graph. The context c provides the global structure and feature information of the graph which is updated by learning W. We suppose that the node similar to the global context should get higher attention weights. To make the attention a_i aware of c, we calculate the inner product of c and each node embedding u_i. To ensure the attention weights are in the range $[0, 1]$, we apply a sigmoid function on a_i. Finally, we acquire the graph embedding e by the weighted sum of node embeddings as

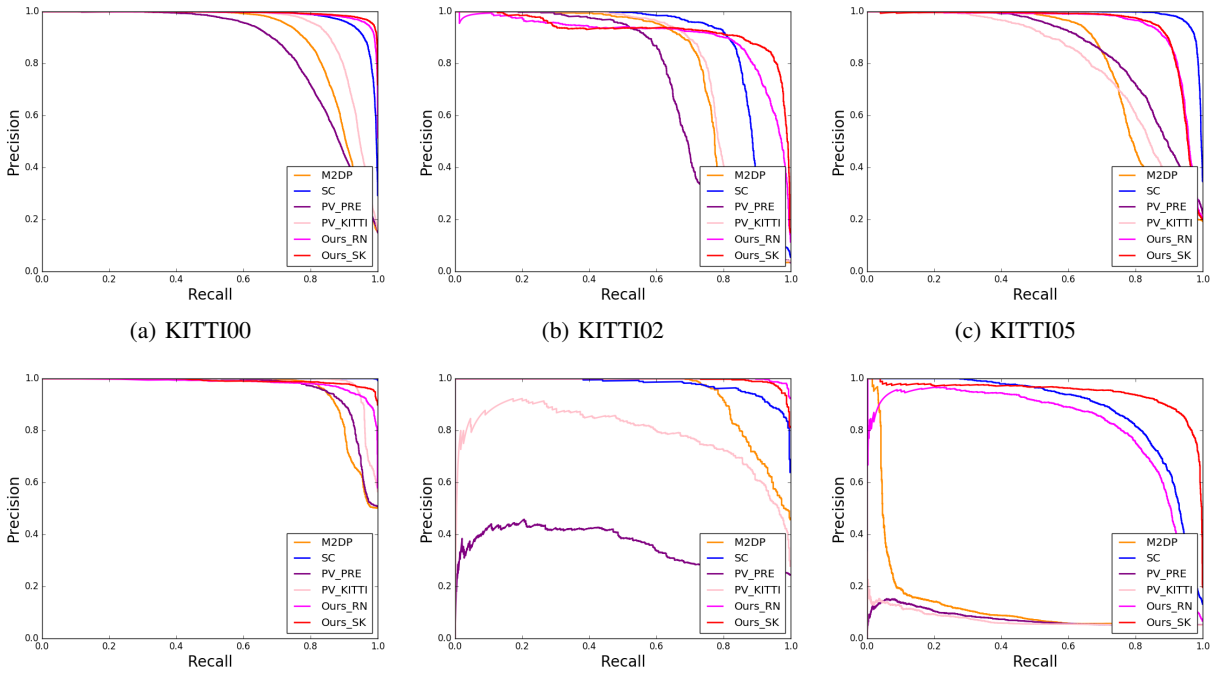
$$\begin{aligned} e &= \sum_{i=1}^{N} sigmoid\left(a_i\right) u_i \\ &= \sum_{i=1}^{N} sigmoid\left(u_i tanh\left(\left(\frac{1}{N}\sum_{m=1}^{N} u_m\right) W\right)^T\right) u_i. \end{aligned} \tag{3}$$

Graph-Graph Interaction: The relation of a pair of graph-level embeddings e_1, e_2 can be accurately estimated by a Neural Tensor Network (NTN) presented in [40]. The NTN adopts a bilinear tensor layer instead of standard linear neural network layer that directly relates the two vectors across multiple dimensions, which is a more reasonable way than calculating the inner product of e_1 and e_2. As shown in Fig. 2, the model computes a feature vector to measure the relation between graph-level embeddings using a function as

$$g\left(e_1, e_2\right) = ReLU\left(e_1^T \omega^{[1:S]} e_2 + \alpha \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} + b\right), \tag{4}$$

where $g \in \mathbb{R}^S$ is the output of the network. e_1, $e_2 \in \mathbb{R}^D$ are the feature embedding of two graphs. $\omega^{[1:S]} \in \mathbb{R}^{D\times D\times S}$ represents a weight tensor, $\alpha \in \mathbb{R}^{S\times 2D}$ represents a weight vector and $b \in \mathbb{R}^S$ represents a bias. The hyper-parameter S denotes the number of slices and is set to 16.

Graph Similarity: We apply a set of fully connected layers to gradually reduce the dimension of the similarity vector and finally get one score per pair in the range $[0, 1]$. The ground truth is either 0 or 1 as we simplify the problem to a binary classification task. We train the model with a binary cross-entropy loss function.



(a) KITTI00 (b) KITTI02 (c) KITTI05

(d) KITTI06 (e) KITTI07 (f) KITTI08

Fig. 5: Precision-Recall curves on KITTI dataset.

IV. Experiments

A. Dataset and Implementation Details

We evaluate the proposed method over KITTI odometry dataset [16], which contains 11 sequences (from 00 to 10) obtained by a 64-ring LiDAR (Velodyne HDL-64E) with ground-truth poses. The ground-truth poses are used to determine if there is a loop closure. In our experiment, two point cloud scenes consist of a positive pair if the Euclidean distance between them is less than 3 m, while the negative one is over 20 m. Note that in the evaluation, positive pairs with a timestamp greater than 30 s are considered to be the true loop closures. In this setting, easy positive pairs (adjacent scenes) will not be evaluated thus it can reflect the real performances of the algorithms. These sequences (00, 02, 05, 06, 07, and 08) with loop closures are evaluated. The sequence 08 has reverse loops and others only have loop closures in the same direction.

The SemanticKITTI [28] dataset has 28 categories and we merge them to 12 categories as shown in Fig 3. The number of nodes varies in different scenarios, ranging from 10-70 on the KITTI odometry dataset. In the node embedding part, we set $k = 10$ in kNN and pad fake nodes with zero embeddings to obtain a fixed number of nodes (in our experiment, we set it to 100), thus we can train the model with a batch operation. We use 1-fold cross-validation and each sequence is considered as a fold, that is, consider one sequence as a test set and the others as training sets. All the experiments are implemented based on PyTorch [41] and Adam optimizer [42] with a learning rate of 0.001 is used for training. There are a large number of negative pairs, thus we reserve all positive pairs and randomly sample some proportion of negative ones.

Methods	00	02	05	06	07	08	Mean
M2DP [9]	0.836	0.781	0.772	0.896	0.861	0.169	0.719
SC [23]	0.937	0.858	**0.955**	**0.998**	0.922	0.811	0.914
PV-PRE [10]	0.785	0.710	0.775	0.903	0.448	0.142	0.627
PV-KITTI [10]	0.882	0.791	0.734	0.953	0.767	0.129	0.709
Ours-RN	0.960	0.859	0.897	0.944	**0.984**	0.783	0.904
Ours-SK	**0.969**	**0.891**	0.905	0.971	0.967	**0.900**	**0.934**

TABLE I: F_1 max scores on KITTI dataset.

B. Place Recognition Performance

To evaluate our semantic graph representation and graph similarity network, we use both the RangeNet++ predictions from model darknet53[1] (Ours-RN) and SemanticKITTI label (Ours-SK) as front-end, comparing with M2DP[2], Scan Context[3] (SC) and PointNetVLAD. Specifically, for PointNetVLAD, we use both its pretrained (refined version) model[4] (PV-PRE) and retrained model on KITTI (PV-KITTI) taking advantage of 1-fold strategy.

Quantitative Results: We analyze the performance using the precision-recall curve in Fig. 5, and we calculate the

[1]https://github.com/PRBonn/lidar-bonnetal/tree/master/train/tasks/semantic

[2]https://github.com/LiHeUA/M2DP

[3]https://github.com/irapkaist/scancontext

[4]https://drive.google.com/file/d/1wYsJmfd2yfbK9DHjFHwEeU1a_x35od61/view

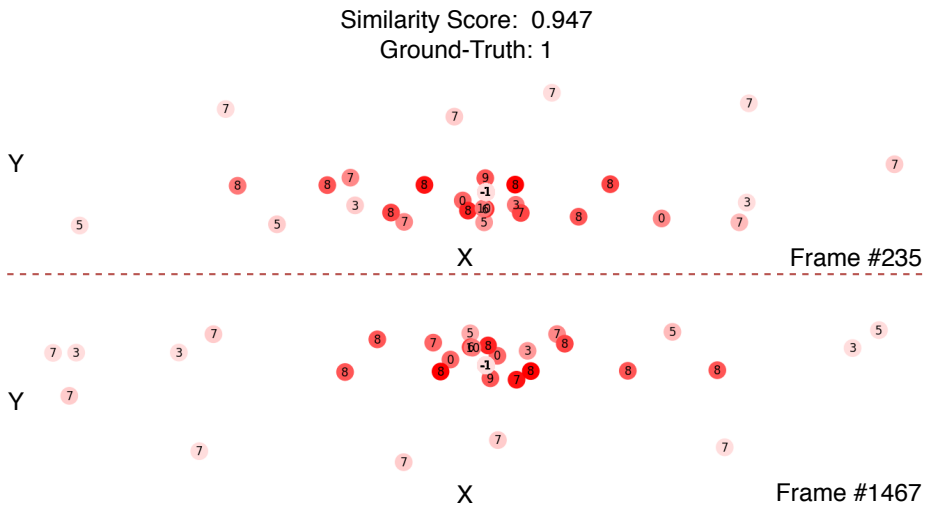


Fig. 6: Graph visualization with attention weights from KITTI08. Note that it's the same place with opposite views. The number denotes the category of each node, and the color intensity represents the attention weight. The deeper the color, the higher the weight. Fake nodes (zero-padding nodes) are -1 in figure and their one-hot encoding is zero vector.

Methods	Occlusion								Rotation							
	00	02	05	06	07	08	Mean	Cmp	00	02	05	06	07	08	Mean	Cmp
M2DP [9]	0.251	0.128	0.324	0.669	0.549	0.102	0.337	-0.382	0.425	0.344	0.415	0.668	0.590	0.348	0.465	-0.245
SC [23]	0.916	**0.847**	**0.925**	**0.996**	0.850	0.721	0.876	-0.038	0.937	0.859	**0.954**	**0.998**	0.936	0.813	0.916	**+0.002**
PV-PRE [10]	0.664	0.610	0.661	0.813	0.439	0.169	0.560	-0.067	0.332	0.133	0.348	0.668	0.647	0.202	0.388	-0.239
PV-KITTI [10]	0.777	0.696	0.632	0.900	0.579	0.112	0.619	-0.090	0.253	0.132	0.713	0.670	0.435	0.156	0.393	-0.316
Ours-RN	0.935	0.817	0.862	0.932	0.928	0.754	0.871	**-0.033**	0.959	0.858	0.894	0.939	**0.977**	0.779	0.901	-0.003
Ours-SK	**0.941**	0.841	0.864	0.954	**0.935**	**0.844**	**0.897**	-0.037	**0.968**	**0.892**	0.902	0.966	0.965	**0.903**	**0.933**	-0.001

TABLE II: F_1 max scores on KITTI dataset when the point clouds are randomly occluded with 30° and rotated around z-axis. Cmp is the comparison with the standard results shown in Table. I

maximum value of F_1 score to evaluate different precision-recall curves shown in Table. I. F_1 score is defined as

$$F_1 = 2 \times \frac{P \times R}{P + R}, \tag{5}$$

where P denotes precision and R denotes recall. As shown in Fig. 5 and Table. I, our mean F_1 max score outperforms other existing methods and our overall performance is competitive. Especially for the challenging sequence 08 with reverse loops, M2DP, PV-PRE, and PV-KITTI have severe degradation. Such methods based on global descriptors cannot handle the viewpoint variations, while our approach performs consistently. Thus when the viewpoint changes, we can still report a confident result, which is further proved in Section IV-C. Notably, the IoU (intersection-over-union) of RangeNet++ is only 52%, which is not high and will introduce noises like wrong labels and centroid offset. Though Ours-RN is indeed lower than Ours-SK, it performs satisfactorily, which indicates that better semantic prediction will bring improvements. The evaluation results demonstrate that our method is effective in large-scale place recognition.

Qualitative Results: Visualizations of the node attentions in graph embedding is shown in Fig. 6. We observe that nodes closer to the center receive higher attention while fake nodes (the zero-padding nodes are -1 in figure and their one-hot encoding is zero vector) receive lower attention. These results are of intuitive significance and further prove the effectiveness of the proposed method.

Furthermore, we show a piece of qualitative visualization results in Fig. 7. We randomly choose a specific single scene and measure similarities between it with other scenes in each sequence. Then, we color the trajectories according to scores. In fact, only the scenes near this chosen scene within a distance threshold (e.g. 3 m) should be similar (color closer to purple). We find our predictions are distinct and accurate.

C. Robustness Test

Occlusion: In a real scenario, dynamic objects (person, vehicle, etc.) inevitably occur during long-term localization, which brings occlusion in LiDAR point clouds. For validation, we randomly select a certain angle in the azimuth and remove points within this area. In practice, we report results with a removal range of 30° in Table. II. All methods have some performance degradation due to information loss, while our methods are the least affected. Because M2DP depends on the point projection, it is sensitive to the change of point distribution caused by the occlusion. PointNetVLAD randomly samples 4096 points per submap as its input,

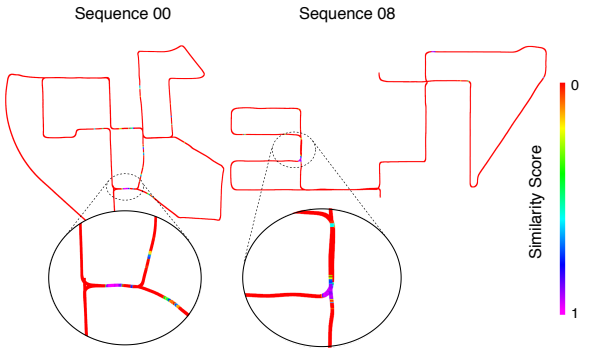


Fig. 7: Similarity visualization. We randomly choose a single scene in each sequence and zoom in similarity scores around this scene. (Best viewed with zoom-in.)

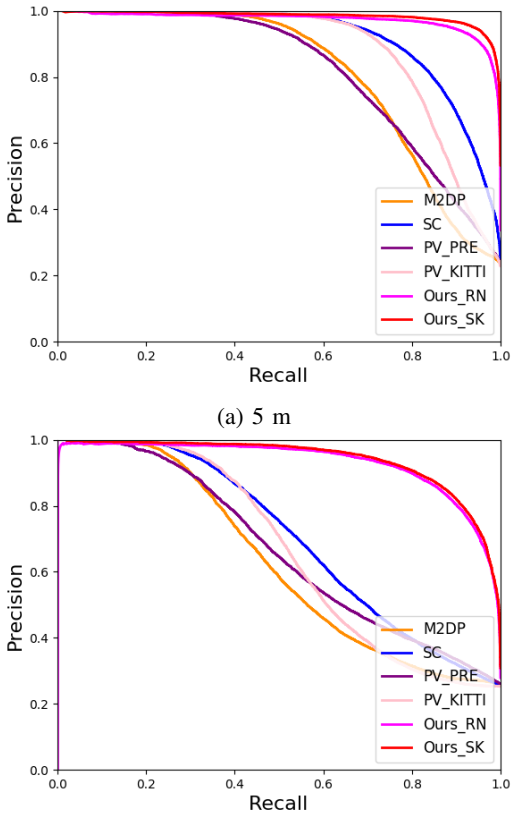
resulting in vulnerability to the absence of the raw point clouds. Scan Context divides the scene into several bins and the miss of a few bins has less impact. The occlusion in our semantic graph representation is equivalent to the disappearance of some nodes, which is only a small part of our representation, and the graph similarity network is robust against the missing to some extent.

Viewpoint Changes: In practice, the viewpoint often changes when arriving at the same place. Thus it's crucial to handle the viewpoint changes. We randomly rotate the point cloud's heading and the results are shown in Table. II. Due to the lack of rotation invariance, the effectiveness of most methods drops dramatically. Although PointNetVLAD has the T-Net modules aiming to regularize the input, it is still not enough to achieve the rotation invariance. Scan Context calculates distances with all possible column-shifted scan contexts and finds the minimum distance, which introduces repeat computation. Compared with local features and distributed features, the semantic information and topological relations among nodes are rotation invariant and our semantic graph representation captures both of them. The spatial and feature relations among nodes are encoded in the node embedding part of our graph similarity network. Besides, compared with the raw point clouds, the number of nodes in the graph is relatively small, which reduces solution space and enables the network easy to converge.

Distance Thresholds: For specific task and application scenario, different distance thresholds of positive pairs for loop closure is needed. To evaluate the adaptability of our approach, we adopt thresholds of 3 m, 5 m, and 10 m to train the corresponding models. The Precision-Recall curves on KITTI00 with 3 m, 5 m and 10 m are shown in Fig. 5a, Fig. 8a and Fig. 8b respectively, which indicates that our approach can maintain satisfactory performance.

D. Efficiency

For each frame, the descriptor size of M2DP, PointNetVLAD, Scan Context and ours is 192, 256, 20×60 and 100×4 (N×f_i), respectively. Our graph similarity network is lightweight, fast, and capable of obtaining similarity scores



(a) 5 m

(b) 10 m

Fig. 8: Precision-Recall curves on KITTI00 with different distance thresholds.

for N places via a single pass through the network instead of comparing them one by one. With the batch size of 128, the inference takes about 9 ms and occupies 2820 MB GPU memory on an NVIDIA GeForce RTX 2080 Ti, making it applicable in real-time robotics systems.

V. Conclusion

In this paper, we propose a novel semantic graph based approach for large-scale place recognition in 3D point clouds, providing a promising direction for future research for a more complete exploitation of semantic information for place recognition. Compared to the existing methods focus on extracting local, global, and statistical features of raw point clouds, acting at the semantic level offers several advantages in environmental changes and is more closely to the way humans perceive scenes. Exhaustive evaluations demonstrate the feasibility and robustness of our approach, especially for reverse loops.

In future work, we will investigate in unsupervised semantic feature learning of point clouds and its availability on place recognition.

References

[1] M. Cummins and P. Newman, "Fab-map: Probabilistic localization and mapping in the space of appearance," *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 647–665, 2008.

[2] M. J. Milford and G. F. Wyeth, "Seqslam: Visual route-based navigation for sunny summer days and stormy winter nights," in *2012 IEEE International Conference on Robotics and Automation*, pp. 1643–1649, IEEE, 2012.

[3] T. Sattler, M. Havlena, K. Schindler, and M. Pollefeys, "Large-scale location recognition and the geometric burstiness problem," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1582–1590, 2016.

[4] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 2, pp. 2169–2178, IEEE, 2006.

[5] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *CVPR 2010-23rd IEEE Conference on Computer Vision & Pattern Recognition*, pp. 3304–3311, IEEE Computer Society, 2010.

[6] M. Dixit, S. Chen, D. Gao, N. Rasiwasia, and N. Vasconcelos, "Scene classification with semantic fisher vectors," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2974–2983, 2015.

[7] M. D. Dixit and N. Vasconcelos, "Object based scene representations using fisher scores of local subspace projections," in *Advances in Neural Information Processing Systems*, pp. 2811–2819, 2016.

[8] S. Lowry, N. Sünderhauf, P. Newman, J. J. Leonard, D. Cox, P. Corke, and M. J. Milford, "Visual place recognition: A survey," *IEEE Transactions on Robotics*, vol. 32, no. 1, pp. 1–19, 2015.

[9] L. He, X. Wang, and H. Zhang, "M2dp: A novel 3d point cloud descriptor and its application in loop closure detection," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 231–237, IEEE, 2016.

[10] M. Angelina Uy and G. Hee Lee, "Pointnetvlad: Deep point cloud based retrieval for large-scale place recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4470–4479, 2018.

[11] Z. Liu, S. Zhou, C. Suo, P. Yin, W. Chen, H. Wang, H. Li, and Y.-H. Liu, "Lpd-net: 3d point cloud learning for large-scale place recognition and environment analysis," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2831–2840, 2019.

[12] Z. Liu, C. Suo, S. Zhou, F. Xu, H. Wei, W. Chen, H. Wang, X. Liang, and Y.-H. Liu, "Seqlpd: Sequence matching enhanced loop-closure detection based on large-scale point cloud description for self-driving vehicles," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1218–1223, IEEE, 2019.

[13] R. Dubé, D. Dugas, E. Stumm, J. Nieto, R. Siegwart, and C. Cadena, "Segmatch: Segment based place recognition in 3d point clouds," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5266–5272, IEEE, 2017.

[14] R. Dubé, A. Cramariuc, D. Dugas, J. Nieto, R. Siegwart, and C. Cadena, "SegMap: 3d segment mapping using data-driven descriptors," in *Robotics: Science and Systems (RSS)*, 2018.

[15] R. Dubé, A. Cramariuc, D. Dugas, H. Sommer, M. Dymczyk, J. Nieto, R. Siegwart, and C. Cadena, "Segmap: Segment-based mapping and localization using data-driven descriptors," *The International Journal of Robotics Research*, p. 0278364919863090, 2019.

[16] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[17] A. E. Johnson and M. Hebert, "Using spin images for efficient object recognition in cluttered 3d scenes," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 21, no. 5, pp. 433–449, 1999.

[18] M. Bosse and R. Zlot, "Place recognition using keypoint voting in large 3d lidar datasets," in *2013 IEEE International Conference on Robotics and Automation*, pp. 2677–2684, IEEE, 2013.

[19] W. Wohlkinger and M. Vincze, "Ensemble of shape functions for 3d object classification," in *2011 IEEE international conference on robotics and biomimetics*, pp. 2987–2992, IEEE, 2011.

[20] Y. Wang, Z. Sun, J. Yang, and H. Kong, "Lidar iris for loop-closure detection," *arXiv preprint arXiv:1912.03825*, 2019.

[21] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 652–660, 2017.

[22] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "Netvlad: Cnn architecture for weakly supervised place recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5297–5307, 2016.

[23] G. Kim and A. Kim, "Scan context: Egocentric spatial descriptor for place recognition within 3d point cloud map," *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4802–4809, 2018.

[24] B. Wu, A. Wan, X. Yue, and K. Keutzer, "Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud," *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1887–1893, 2017.

[25] B. Wu, X. Zhou, S. Zhao, X. Yue, and K. Keutzer, "Squeezesegv2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a lidar point cloud," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 4376–4382, IEEE, 2019.

[26] X. Kong, G. Zhai, B. Zhong, and Y. Liu, "Pass3d: Precise and accelerated semantic segmentation for 3d point cloud," *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3467–3473, 2019.

[27] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss, "Rangenet++: Fast and accurate lidar semantic segmentation," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2019.

[28] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, "Semantickitti: A dataset for semantic scene understanding of lidar sequences," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 9297–9307, 2019.

[29] H. Yin, L. Tang, X. Ding, Y. Wang, and R. Xiong, "Locnet: Global localization in 3d point clouds for mobile vehicles," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 728–733, IEEE, 2018.

[30] H. Bunke, "What is the distance between graphs," *Bulletin of the EATCS*, vol. 20, pp. 35–39, 1983.

[31] H. Bunke and K. Shearer, "A graph distance metric based on the maximal common subgraph," *Pattern recognition letters*, vol. 19, no. 3-4, pp. 255–259, 1998.

[32] Z. Zeng, A. K. Tung, J. Wang, J. Feng, and L. Zhou, "Comparing stars: On approximating graph edit distance," *Proceedings of the VLDB Endowment*, vol. 2, no. 1, pp. 25–36, 2009.

[33] D. B. Blumenthal and J. Gamper, "On the exact computation of the graph edit distance," *Pattern Recognition Letters*, 2018.

[34] Y. Bai, H. Ding, S. Bian, T. Chen, Y. Sun, and W. Wang, "Simgnn: A neural network approach to fast graph similarity computation," in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pp. 384–392, ACM, 2019.

[35] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations (ICLR)*, 2017.

[36] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 5, p. 146, 2019.

[37] F. Husain, H. Schulz, B. Dellen, C. Torras, and S. Behnke, "Combining semantic and geometric features for object class segmentation of indoor scenes," *IEEE Robotics and Automation Letters*, vol. 2, no. 1, pp. 49–55, 2016.

[38] J. Bao, P. Liu, and S. V. Ukkusuri, "A spatiotemporal deep learning approach for citywide short-term crash risk prediction with multi-source data," *Accident Analysis & Prevention*, vol. 122, pp. 239–254, 2019.

[39] X. Geng, Y. Li, L. Wang, L. Zhang, Q. Yang, J. Ye, and Y. Liu, "Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 3656–3663, 2019.

[40] R. Socher, D. Chen, C. D. Manning, and A. Ng, "Reasoning with neural tensor networks for knowledge base completion," in *Advances in neural information processing systems*, pp. 926–934, 2013.

[41] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, pp. 8024–8035, 2019.

[42] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.