Retriever: Point Cloud Retrieval in Compressed 3D Maps

Louis Wiesmann Rodrigo Marcuzzi Cyrill Stachniss Jens Behley

***Abstract*— Most autonomous driving and robotic applications require retrieving map data around the vehicle's current location. Those maps can cover large areas and are often stored in a compressed form to save memory and allow for efficient transmission. In this paper, we address the problem of place recognition in a compressed point cloud map. To this end, we propose a novel deep neural network architecture that directly operates on a compressed feature representation produced by a compression encoder. This enables us to bypass compute-heavy decompression of the map and exploits the compact as well as descriptive nature of the compressed features. Additionally, we propose an alternative to the commonly used NetVLAD layer to aggregate local descriptors. Here, we utilize an attention mechanism between local features and a latent code. Our experiments suggest that this produces a more descriptive feature representation of the point clouds for place recognition. We experimentally validate all architectural choices we made by our ablation studies and compare our performance to other state-of-the-art baselines on two commonly used datasets.**

I. Introduction

The ability to localize in a map is a key ingredient of most robotic systems and autonomous cars. For this purpose, these systems require a map of their surroundings, which can have different representations, like grid maps [23], [16], semantic maps [7], [41], mesh-based maps [8], [46], or point clouds [13]. In place recognition, we want to determine if the current place, the so-called query, has been visited before. Ideally, we want furthermore to retrieve the corresponding match from the map for localization. It is therefore crucial to find a representation of that data, i.e., queries and maps (database), and a suitable similarity measure that allows comparing the queries with the database entries.

To solve the place recognition problem in point cloud maps, usually one compares either directly the point clouds or utilizes extracted descriptors. Those descriptors are commonly based on local features that are aggregated into histograms or feature vectors [38], [37], [39], [20]. While classical approaches used mainly handcrafted features, recent approaches increasingly rely on learning-based features [21], [4], [52], [6]. The descriptors are a compact and descriptive representation of the maps but do not allow for reconstruction or can be used to solve other tasks. For instance, scan registration for more fine-grained localization or change detection still requires storing the original data. Especially in autonomous driving, those maps cover large areas and therefore require a substantial amount of memory [17], which makes compression necessary for storage and transmission. Localizing in compressed maps have been exploited for 2D grid maps [50] but not yet for 3D point clouds.

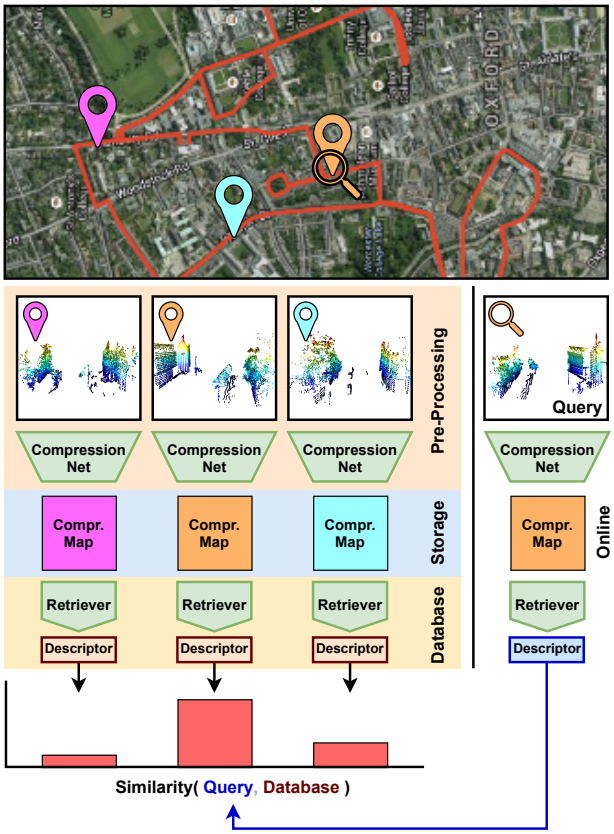

Fig. 1: Point Cloud-based place recognition. We compress the point clouds from a map using a compression network. The resulting compressed encodings are used for decompression, transmission, or place recognition, such that the original memory-consuming point clouds do not need to be stored. For place recognition, we propose the Retriever which extracts from the compressed representation a descriptor. When revisiting an area, one can retrieve the corresponding map by comparing the descriptors of the current position (query) and the compressed descriptors in a database.

The main contribution of this paper is a LiDAR-based place recognition approach that directly operates on compressed point clouds. It exploits the urge for a memory-efficient representation needed for storing and transmitting large-scale point clouds in two aspects. First, this enables us to bypass the compute-heavy decompression. Second, it utilizes the compact and descriptive nature of our compressed feature representation to tackle the place recognition prob-

All authors are with the University of Bonn, Germany. Cyrill Stachniss is additionally with the Department of Engineering Science at the University of Oxford, UK.

This work has partially been funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy, EXC-2070 – 390732324 – PhenoRob and by the European Union's Horizon 2020 research and innovation programme under grant agreement No 101017008 (Harmony).

lem. To this end, we propose a novel neural network called Retriever that consists of three parts. First, an encoder extracts the compact task-agnostic feature representation from the point clouds, which can be used for storage, transmission, decompression [51], or place recognition. Then, we feed the compressed features into a feature propagation network to compute a more refined task-specific representation. Finally, we aggregate the computed features by a novel perceiver-based attention module extending [18] into a global descriptor, which we compare to other descriptors in a database to retrieve the corresponding maps. The workflow of our approach is visualized in Fig. 1.

In sum, we propose a place recognition pipeline that exploits a compressed point cloud representation and a novel attention-based aggregation module. Working directly on the compressed representation allows for bypassing compute-heavy decompression and utilizes the urge of a memory-efficient representation for storage. Our implementation, data, and the pretrained models will be publicly available at `https://github.com/PRBonn/retriever`.

II. Related Work

Place recognition is the task of recognizing previously visited parts of an environment. The most common way to solve the problem is to use images for representing the map and the vehicle's surrounding [48], [47], [34], [27], [29]. This task is often solved in two steps: First, distinct local features are extracted in either a handcrafted [26], [2], [32], [37], [38] or computed in a data-driven learning-based fashion [21], [4], [52], [6]. Second, those local features are often aggregated into a global descriptor, which will then be used for determining similarity between global descriptors of places in the map. Classical approaches utilize bag of visual words (BoW) [36], BoW on intensity data [10], [35], or a vector of locally aggregated descriptors (VLAD) [19] to compute the global descriptor. NetVLAD [1] relaxes the hard assignments of VLAD to soft assignments to make the operation differentiable and fully end-to-end learnable. With the success of NetVLAD, deep learning-based methods also evolved in different domains for place recognition, such as LiDAR-based [44] or RADAR-based place recognition [33].

Nowadays, NetVLAD is a popular building block for many point cloud-based place recognition approaches utilizing deep neural networks [44], [24], [12], [52]. These point cloud-based place recognition networks usually follow the aforementioned two-step paradigm of first computing expressive local features, which then are aggregated into a global descriptor. The local features can be solely learned [12], [44], [52] or enhanced with classical 3D descriptors [24]. A third way to obtain local features is from networks that are pretrained on different tasks and thus produce task-agnostic features [54], [42], [14]. Our approach is one of the latter ones where we utilize a compression network to produce a compact feature representation that can be pretrained self-supervised. This provides us with descriptive features, which are additionally well suited for storage, transmission, and can later still be used for decompression.

3D neural networks often utilize convolutions [43], graphs [22], [40], or point-wise shared MLPs [30], [31] to propagate the features. The attention mechanism of transformers [45] is rarely used in the 3D domain due to the huge memory consumption induced by the self-attention mechanism for the typically large number of 3D input points. However, the attention mechanism has specific properties that make it theoretically well-suited for point clouds: it is permutation invariant, and it propagates features based on the entire input sequence without the need of downsampling.

Some approaches address the memory problem of self-attention by either approximating the self-attention mechanism [49], [9] or applying it only to a subset of the sequence [11], [57]. The Perceiver [18] tackles the problem by completely bypassing self-attention on the input representation. It uses cross-attention between the input sequence and several latent feature vectors that are learned while training. The self-attention mechanism is only used on the fewer latent feature vectors.

In contrast to the aforementioned point cloud-based place recognition approaches, we use the Perceiver idea for aggregating local features produced in the first stage into a global descriptor and thereby substituting the commonly used NetVLAD layer. The local features are provided by task-agnostic features from an encoder that was originally designed for point cloud compression [51].

III. Our Approach

In this work, we tackle the problem of place recognition in 3D point clouds. The goal is to compute a single global descriptor for each point cloud such that spatially nearby point clouds have similar descriptors while being dissimilar to point clouds from different places.

To this end, we propose an approach that tackles the problem in two steps, as depicted in Fig. 2. We first compute expressive local features, which we aggregate in a second step to a global descriptor. The local features are computed by a convolutional encoder followed by a feature propagation module. For computing the global descriptor, we propose an attention-based feature aggregation network. The place recognition task can then be solved by comparing the descriptors of the query with the ones in the database.

By utilizing a point cloud compression encoder for the feature generation, we have a memory-efficient and task-agnostic intermediate representation, from which we can restore the point cloud and can still later use it for other tasks, which require a map. We are able to operate directly with the compressed representation and, thus, can bypass decompression completely. This is not only memory- and compute-efficient but also provides expressive features for place recognition.

A. Compression Network

Our goal is to retrieve in a compressed point cloud map. For creating this compressed representation in the first place, we use the autoencoder proposed in our previous work [51]. The encoder consists of three ResNet-KPConv blocks [43]

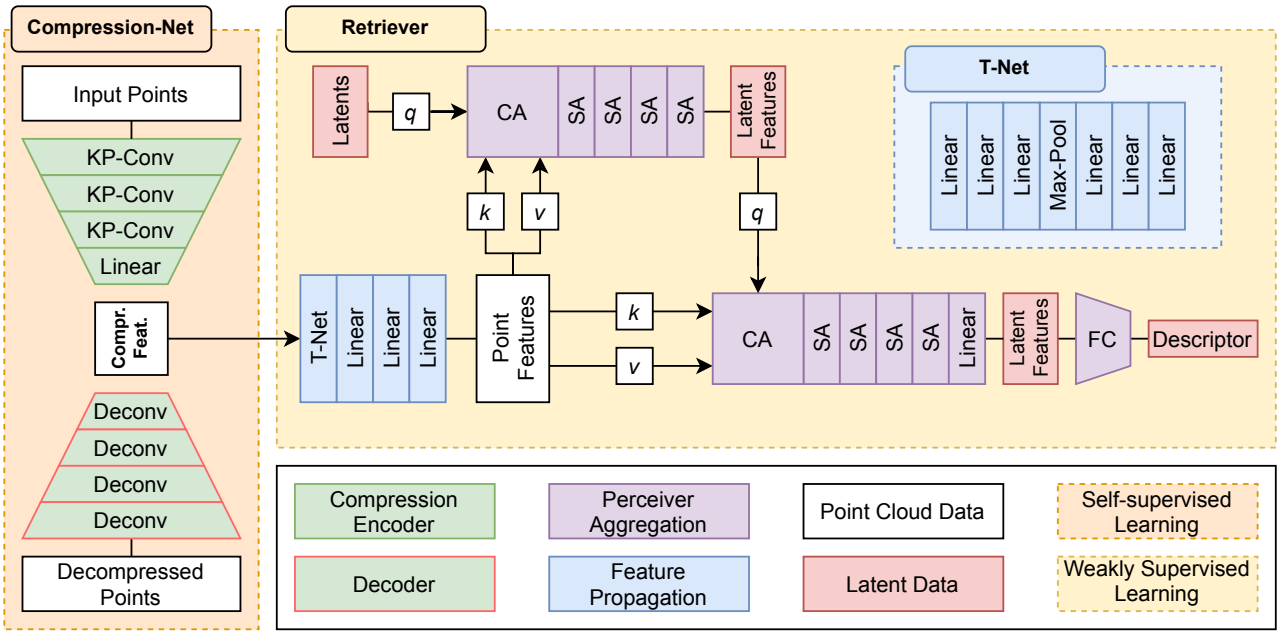

Fig. 2: Our proposed architecture for point cloud-based place recognition. The Compression-Net (green) computes a compact feature representation and stems from our previous work [51]. Those task-agnostic features can be used for efficient storage, transmission, decompression, and place recognition. The feature propagation network (blue) transforms the features into task-specific features suited for place recognition. The Perceiver (purple) aggregation computes latent features by cross attention (CA) between those compressed features and a latent representation. Self-attention (SA) on the resulting latent features refines them and a fully connected layer (FC) finally aggregates them into a global descriptor. Note that the decoder of the Compression-Net is only used for self-supervised pretraining and decompression.

with an additional feature compression projection head and a decoder with four deconvolutional blocks. For a given input point cloud $\boldsymbol{P} \in \mathbb{R}^{N\times 3}$, the encoder $E : \mathbb{R}^{N\times 3} \mapsto \mathbb{R}^{N_c \times D_c}$ computes a small subset $N_c \ll N$ of points, with an expressive feature representation $\boldsymbol{F}_c = E(\boldsymbol{P})$ from which the decoder can recover the dense point cloud. Many approaches use local, neighborhood-aware features to improve the place recognition performance [37], [38]. Having features that can reconstruct the local surrounding motivated us to conduct the research described here as those features should be well suited and discriminative for place recognition. This allows us to directly operate on the compressed representation itself and does not require decompression to regain the point cloud information or computation of other local features. Since saving large-scale point cloud maps in a compressed format is often anyway needed [17], being able to work with the compressed representation directly saves compute and memory.

We pretrain the network as described in the original paper [51] using a self-supervised reconstruction task, where the employed auto encoder should reconstruct the input as faithfully as possible. For more details on the training, we refer to the original paper [51].

Utilizing self-supervised pretraining has the advantage of having easy access to a lot more data, and therefore it is usually possible to generate generalizable feature representations, which are less prone to over-fitting [3], [53], [56]. Additionally, by freezing the encoder while learning place recognition, we keep the ability to reconstruct the point cloud needed for other downstream tasks by using the decoder for decompression.

B. Feature Propagation Network

The compression network extracts a smaller subset of points with a feature representation from which the point cloud can be reconstructed. Having such an expressive representation does, however, not mean that it is also the best representation for place recognition. For example, the compressed feature representation cannot be rotational invariant since it must be able to decompress the point cloud with the correct orientation.

For place recognition, we aim for a descriptor that is independent of the viewpoint or driving direction. Therefore, we use a PointNet [30] variant, which transforms each point feature in a high-dimensional nonlinear space. Our PointNet variant consists of a TNet followed by an MLP. The T-Net : $\mathbb{R}^{N_c\times D_c} \mapsto \mathbb{R}^{D_c^2}$, computes a transformation $\boldsymbol{T} \in \mathbb{R}^{D_c\times D_c}$ based on the compressed features $\boldsymbol{F}_c \in \mathbb{R}^{N_c\times D_c}$. It consists of a multi-layer perceptron[1] MLP$(64, 128, 1024)$, which is followed by global max pooling and a second MLP$(512, 256, D_c^2)$ that transforms the extracted descriptor to the dimensionality of the desired transformation. We apply the transformation $\boldsymbol{T}$ to the input features

$$\boldsymbol{F}'_c = \boldsymbol{F}_c(\boldsymbol{I}_d + \boldsymbol{T}), \tag{1}$$

[1]Here, we use the convention that the argument of MLP correspond to the number of channels of the output, e.g., MLP$(4, 16)$ takes a not further specified input and produces 4 and then 16 output channels in the intermediate and final feature map.

where $\boldsymbol{I}_d$ corresponds to the identity matrix and therefore $\boldsymbol{T}$ is just a residual added to the identity that facilitates learning. This provides the network with the possibility to extract for each point cloud a specific transformation to transform it into a common frame to achieve transformation invariance.

A following MLP$(64, 128, 512)$ projects the transformed features $\boldsymbol{F}'_c \in \mathbb{R}^{N_c \times 6}$ of the T-Net into a higher-dimensional feature space $\boldsymbol{F}_p \in \mathbb{R}^{N_c \times 512}$ that is specialized for place recognition.

C. Perceiver Aggregation

In the two previous parts, we have described how to compute a set of local features $\boldsymbol{F}_p$ from the source point cloud. However, our goal is to have exactly one global descriptor $\boldsymbol{z} \in \mathbb{R}^{D_g}$ for each point cloud, which can be used for place recognition by computing a descriptor similarity. Consequently, we need to aggregate the local features into a global descriptor. The original PointNet paper [30] uses global max pooling, while in the place recognition domain many approaches [24], [44], [55] utilize the more sophisticated NetVLAD [1] layer for aggregating the features.

In this work, we propose a novel attention-based aggregation method based on the Perceiver [18], which is a variant of the Transformer [45]. The attention mechanism in the Transformers computes new features $\boldsymbol{F}_t \in \mathbb{R}^{N_t \times D}$ by a linear combination of a set of value vectors $\boldsymbol{V} \in \mathbb{R}^{N_s \times D}$. The weighting $\boldsymbol{W} \in \mathbb{R}^{N_t \times N_s}$ of the features depend on the outer cross product of the queries $\boldsymbol{Q} \in \mathbb{R}^{N_t \times D}$ and the keys $\boldsymbol{K} \in \mathbb{R}^{N_s \times D}$

$$\boldsymbol{F}_t = \boldsymbol{W}\boldsymbol{V} = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{D}}\right)\boldsymbol{V}, \tag{2}$$

where the keys $\boldsymbol{K} = \boldsymbol{W}_k\boldsymbol{X}_s$ and values $\boldsymbol{V} = \boldsymbol{W}_v\boldsymbol{X}_s$ are linear projections of the source sequence $\boldsymbol{X}_s$, while the queries $\boldsymbol{Q} = \boldsymbol{W}_v\boldsymbol{X}_t$ are projections from the target sequence $\boldsymbol{X}_t$. The softmax ensures that the weights of the linear combinations sum up to one. For the case of self-attention, where $\boldsymbol{X}_s = \boldsymbol{X}_t$ and thus $N_s = N_t$, the weight matrix $\boldsymbol{W}$ grows quadratically with respect to the sequence length. Since point clouds usually have thousands to millions of points, this is too memory expensive for most modern GPUs.

Instead of doing self-attention on the input sequence, the Perceiver [18] solves the problem by, using a few latent vectors as target sequence $\boldsymbol{X}_t \in \mathbb{R}^{N_t \times D}$ with $N_t \ll N_s$. These latent vectors are learned and optimized while training. Multiple self-attention layers use a cross-attention block between the input sequence and the latent vectors on the latent features $\boldsymbol{F}$. Since N_t is a constant and not depending on the input sequence length N_s, the computational complexity and memory consumption decreases from $\mathcal{O}(N_s^2)$ to $\mathcal{O}(N_s)$.

In our application, we use the features coming from our feature propagation network as input sequence for the Perceiver $P : \mathbb{R}^{N_s \times D_{\text{in}}} \mapsto \mathbb{R}^{N_t \times D_{\text{out}}}$. Our Perceiver uses two cross attention blocks for propagating the input features to the latent vectors, where each cross attention block is followed by 4 self-attention blocks working solely on the latent features. All perceiver blocks are implemented as ResNet blocks [15]. A fully connected layer projects the latent features in the end to the desired output dimension of the global descriptor $\boldsymbol{z} \in \mathbb{R}^{D_g}$:

$$\boldsymbol{z} = \boldsymbol{W}_g\boldsymbol{f} + \boldsymbol{b}_g, \tag{3}$$

with $\boldsymbol{W}_g \in \mathbb{R}^{D_g \times N_t \cdot D_{\text{out}}}, \boldsymbol{b}_g \in \mathbb{R}^{D_g}$. Note that each operation is permutation invariant, which makes it perfectly suited for the unordered nature of point clouds. As the Perceiver always produces a fixed output feature independent of the number of inputs, we are able to process point clouds of arbitrary sizes.

Both, our Perceiver and the NetVLAD layer aggregate local features relative to a common global context. This feature representation is, in the case of the NetVLAD layer, a set of centroids that are learned while training. For the feature aggregation, they accumulate for each centroid the residual to each input feature weighted by their reciprocal squared distances.

The Perceiver uses the latent vectors as global context. Instead of accumulating the residuals, it recombines the input features using the cross attention mechanism. The NetVLAD stops at this point with the feature propagation and uses a fully connected layer for aggregating the global descriptor. Our Perceiver uses multiple self-attention and cross-attention blocks allowing it to propagate information also between the latent features for a more refined representation.

D. Loss Function

We use the Lazy quadruplet loss proposed by Uy et al. [44] to optimize our architecture for place recognition. Given a query descriptor $\boldsymbol{z}_q \in \mathbb{R}^{D_g}$ as well as a set of positive $\mathcal{Z}^+ = \{\boldsymbol{z}_1^+, \ldots, \boldsymbol{z}_{N^+}^+\}$ and negative $\mathcal{Z}^- = \{\boldsymbol{z}_1^-, \ldots, \boldsymbol{z}_{N^-}^-\}$ examples, the lazy quadruplet loss is defined as

$$\mathcal{L} = \max(\delta^+ - \delta^- + m_1, 0) + \max(\delta^+ - \delta^* + m_2, 0), \tag{4}$$

where $\delta^+ = ||\boldsymbol{z}_q - \hat{\boldsymbol{z}}^+||_2$ is the Euclidean distance between the query $\boldsymbol{z}_q$ and the hardest positive example $\hat{\boldsymbol{z}}^+$. Consequently, $\delta^- = ||\boldsymbol{z}_q - \hat{\boldsymbol{z}}^-||_2$ is the distance to the hardest negative $\hat{\boldsymbol{z}}^-$ and $\delta^* = ||\hat{\boldsymbol{z}}^- - \hat{\boldsymbol{z}}^*||_2$ is the distance between the hardest negative and a second negative $\hat{\boldsymbol{z}}^* \in \mathcal{Z}^-$. The second negative $\hat{\boldsymbol{z}}^*$ is not only far away from the query $\boldsymbol{z}_q$ but also from all other negatives in $\mathcal{Z}^-$. By this, the loss minimizes the distance between positive pairs and tries to maximize the distance to the negative examples. The second negative is used to keep the distance from other negatives that are also structurally dissimilar.

IV. Implementation Details and Training

In this section, we report the implementation details and the training procedure. We pretrain the compression encoder following the self-supervised training schedule as originally proposed in our former work [51]. We freeze the weights of the encoder, which allows us to keep the ability to use the features also for decompression. Additionally, we can preprocess the point clouds and use bigger batch sizes due to the smaller memory size of the individual point clouds. For

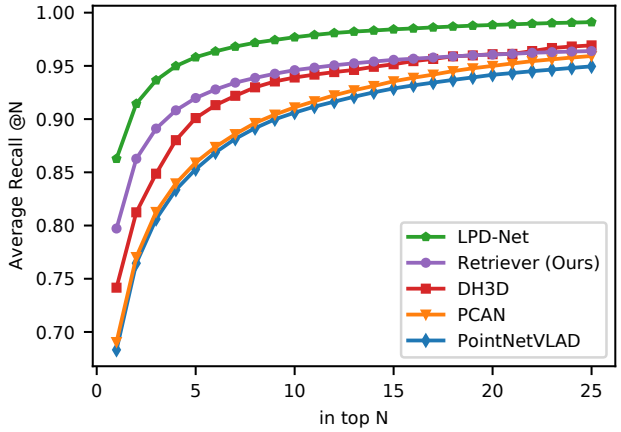

Fig. 3: Average recall @N on the Oxford Robocar dataset. Even though our approach is working on the compressed feature representation, it is able to outperform most of the other baselines that operate on the full input information *without* a memory bottleneck.

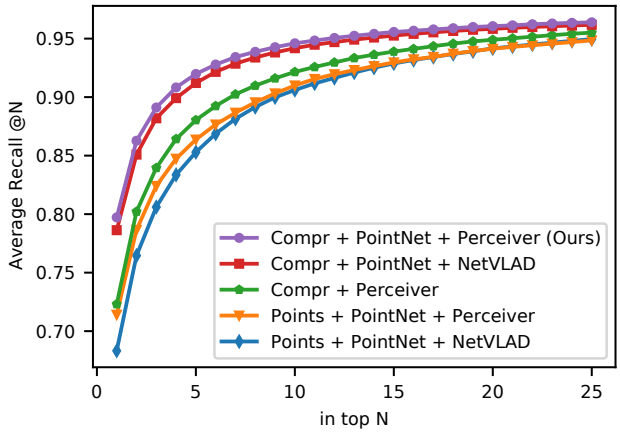

Fig. 4: Average recall @N on the Oxford Robocar dataset for diffenrent network configurations. Replacing the raw input point cloud (Points) by our compressed feature representation (Compr) increases the retrieval accuracy. Similarly, exchanging the NetVLAD layer by our Perceiver-based aggregation module allows also for better place recognition.

training the rest of the network, we use ADAMW [25] with a learning rate of 0.001 and weight decay of 0.01. In the loss, we use the margins $m_1 = 0.5$ and $m_2 = 0.2$. We use a batch size of 8 in all our experiments and use $N^+ = 2$ positive and $N^- = 18$ negative examples for the lazy quadruplet loss. Our approach is implemented and tested with Pytorch in the Pytorch Lightning framework. We use PyKeops [5] for an optimized k-nearest neighbor search on the GPU.

V. Experimental Evaluation

The main focus of this work is to realize 3D point cloud-based place recognition. We present our experiments to illustrate the capabilities of our method and to show that our approach is capable of doing point cloud-based place recognition by exploiting memory-efficient task-agnostic features, which are finally aggregated to a global descriptor by our novel perceiver-based attention module. We evaluate our approach on the widely used Oxford Robocar dataset [28] and the three In-House datasets [44]. We follow the same processing and evaluation as PointNetVLAD [44] to be consistent with the baselines. They use the average recall @N and @1% for evaluating the success rate of the place recognition. Whenever a query point cloud is retrieved within 25 m, it counts as successful.

A. Place Recognition Results

The first experiment evaluates the performance of our Retriever approach, where we compare with other baselines methods, i.e., PointNetVLAD [44], PCAN [55], DH3D [12][2] and LPD-Net [24]. The results for the Oxford dataset are shown in Fig. 3. Our approach is able to outperform most of the baselines in terms of average Recall. The Retriever has a lower performance than LPD-Net, but it should be noted that all methods under comparison dot not perform a reversible compression. Even though the descriptor is compact for all approaches, it can not be used for reconstructing the point clouds. Consequently, when the map needs to be stored in a compressed form, e.g., for downstream tasks like scan registration, the baselines have to spend additional compute for the decompression and feature computation (like the spectral features for LPD-net). The advantage of our approach is that our network operates directly on the compressed features and does not require decompression.

Method	Oxford	U.S.	R.A.	B.D
PointNetVLAD	85.21	74.80	73.39	71.96
PCAN	83.81	79.05	71.18	66.82
LPD-Net	94.92	96.00	90.46	89.14
Retriever (Ours)	91.93	91.88	87.44	85.53

TABLE I: Average recall @1% on different datasets. All models have only been trained Oxford to show their generalization capabilities. Our approach provides competitive results on all datasets.

In Tab. I, we evaluate the performance on multiple datasets that have different configurations (Oxford Robocar [28] and the three routes from the In-House dataset [44]). Each approach is only trained on the Robocar [28] dataset to evaluate how well the methods generalize. Our approach is able to achieve competitive results on all datasets and does not overfit.

B. Qualitative Results

In Fig. 5, we show qualitative results to provide deeper insides into the behavior of our approach and the challenges of the task. The descriptor of the query (blue) is compared against all descriptors of a different run. Here, we can see the top 3 retrieved point clouds using our method (green denotes positive matches, red are from different locations). The query is successfully retrieved in the top 1. As we can see, the second-best match (top 2) is from a different location, while the other true corresponding point cloud is found only at third. Reasons for this could be that the prominent shape of the roof from the query is not visible in the 3. The top 2,

[2]PointNetVLAD was reimplemented and retrained from scratch which outperforms the results of the original paper. The results from DH3D and PCAN stem from `https://github.com/JuanDuGit/DH3D`,

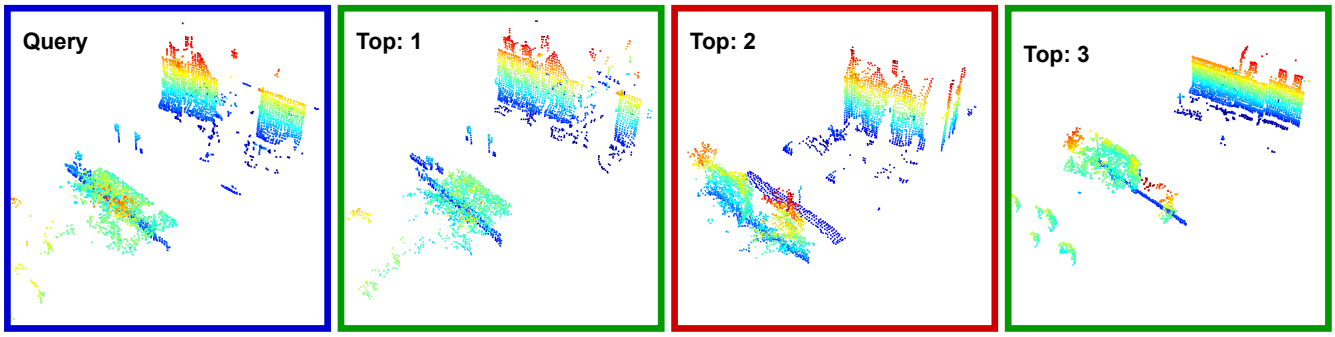

Fig. 5: Qualitative results for a query point cloud (blue) and the top 3 retrieved maps of the Oxford Robocar dataset. Point Clouds with a green frame refer to true corresponding matches, while the red one is from a different position. Our approach is able to find the correct matches (as top 1 and top 3). Sometimes places from different areas (2) are more similar to the query than the true correspondences (3), showing the challanges of the dataset.

however, has a similar roof shape. All point clouds have trees on the opposite side and share thus similar appearance.

C. Ablation Studies

In this section, we validate the choices made in our architecture and evaluate the importance of each part of the network. First, we look at the compressed feature representation produced by the convolutional encoder. After this, we compare our Perceiver-based aggregation module with the current state-of-the-art NetVLAD layer, which is typically used in the place recognition domain. For this, we have trained different networks architectures. The results on the Oxford Robocar dataset are shown in Fig. 4. We either use the original Points or the compressed feature representation (denoted as 'Compr') after the encoder as input to the PointNet block or directly to the Perceiver aggregation. For aggregating the local features to a global descriptor, we either use NetVLAD or our proposed Perceiver-based module.

Compressed Features. As we can see in Fig. 4, using the compressed features in both the NetVLAD aggregation and our aggregation module achieves better results than using the raw input points (compare blue to red and orange to purple). This suggests that using the compressed feature representation is not only advantageous for storage and transmission but also the information of the local neighborhood makes the global descriptors more distinct and mitigates the information loss due to compression.

Perceiver vs. NetVLAD. Exchanging the NetVLAD layer by the Perceiver-based aggregation module increases the performance for the point and compressed feature-based versions (compare Fig. 4 blue to orange and red to purple). The NetVLAD layer aggregates the local features without considering any relation between the features. The attention mechanisms in the Perceiver allow for suppressing unimportant and concentrate more on especially descriptive features. Additionally, the self-attention of the latent features incorporates information from the whole sequence and can thus change the features based on the global context. This could help to describe the point clouds more accurately and, therefore, increase the place recognition performance. Directly aggregating the compressed features to a global descriptor without feeding it to the PointNet yields worse results, showing that transforming the task-agnostic features to a task-specific representation is advantageous (Fig. 4 green and purple).

VI. Discussion

This paper shows that we can reliably retrieve already visited areas in compressed point cloud maps and how the increasingly popular self-attention mechanism for vision-based tasks can be efficiently employed to improve feature aggregation in the domain of LiDAR-based place recognition. A promising avenue for future work would be the investigation of more sophisticated backbones, like the graph neural networks, to further improve the performance as in LPD-Net [24]. Since the Perceiver [18] already shows promising results on different tasks and for different input representations, we believe that our Perceiver aggregation could help improve feature aggregation not only for the point clouds but also in other domains like image-based retrieval.

VII. Conclusion

In this paper, we presented Retriever, a novel approach to tackle 3D point cloud-based place recognition, focusing on full operation on compressed and thus memory-efficient representations. Our method exploits a task-agnostic compact feature representation produced by a compression network. This representation facilitates efficient storage, transmission, decompression, and place recognition. For place recognition, we first translate the compressed features into task-specific, local descriptors, which then are aggregated to a global descriptor. Our proposed aggregation module builds on top of the memory and compute-efficient perceiver architecture. We implemented and evaluated our approach on different datasets and provided comparisons to other existing techniques. The experiments suggest that the compressed feature representation is more memory efficient and more descriptive than working solely on the point cloud representation. Additionally, our perceiver-based aggregation module seems to produce better-suited descriptors than the widely used NetVLAD layer.

REFERENCES

[1] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. NetVLAD: CNN Architecture for Weakly Supervised Place Recognition. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[2] H. Bay, A. Ess, T. Tuytelaars, and L.V. Gool. Speeded-up robust features (SURF). *Journal of Computer Vision and Image Understanding (CVIU)*, 110(3):346–359, 2008.

[3] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging properties in self-supervised vision transformers. *arXiv preprint arXiv:2104.14294*, 2021.

[4] M. Chang, S. Yeon, S. Ryu, and D. Lee. SpoxelNet Spherical Voxel-Based Deep Place Recognition for 3D Point Clouds of Crowded Indoor Spaces. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2020.

[5] B. Charlier, J. Feydy, J.A. Glaunès, F.D. Collin, and G. Durif. Kernel Operations on the GPU, with Autodiff, without Memory Overflows. *Journal of Machine Learning Research*, 22(74):1–6, 2021.

[6] X. Chen, T. Läbe, A. Milioto, T. Röhling, O. Vysotska, A. Haag, J. Behley, and C. Stachniss. OverlapNet: Loop Closing for LiDAR-based SLAM. In *Proc. of Robotics: Science and Systems (RSS)*, 2020.

[7] X. Chen, A. Milioto, E. Palazzolo, P. Giguère, J. Behley, and C. Stachniss. SuMa++: Efficient LiDAR-based Semantic SLAM. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2019.

[8] X. Chen, I. Vizzo, T. Läbe, J. Behley, and C. Stachniss. Range Image-based LiDAR Localization for Autonomous Vehicles. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2021.

[9] K. Choromanski, V. Likhosherstov, D. Dohan, X. Song, A. Gane, T. Sarlos, P. Hawkins, J. Davis, A. Mohiuddin, L. Kaiser, et al. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*, 2020.

[10] L. Di Giammarino, I. Aloise, C. Stachniss, and G. Grisetti. Visual Place Recognition using LiDAR Intensity Information . In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2021.

[11] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[12] J. Du, R. Wang, and D. Cremers. DH3d: Deep Hierarchical 3D Descriptors for Robust Large-Scale 6DoF Relocalization. In *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, pages 744–762, 2020.

[13] R. Dube, M. Gollub, H. Sommer, I. Gilitschenski, R. Siegwart, C. Lerma, and J. Nieto. Incremental segment-based localization in 3d point clouds. *IEEE Robotics and Automation Letters (RA-L)*, 2018.

[14] S. Garg, N. Snderhauf, and M. Milford. Don't look back: Robustifying place categorization for viewpoint and condition-invariant place recognition. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2018.

[15] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[16] A. Hornung, K. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard. OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees. *Autonomous Robots*, 34:189–206, 2013.

[17] L. Huang, S. Wang, K. Wong, J. Liu, and R. Urtasun. Octsqueeze: Octree-structured entropy model for lidar compression. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1313–1323, 2020.

[18] A. Jaegle, F. Gimeno, A. Brock, A. Zisserman, O. Vinyals, and J. Carreira. Perceiver: General perception with iterative attention. *arXiv preprint arXiv:2103.03206*, 2021.

[19] H. Jégou, M. Douze, C. Schmid, and P. Pérez. Aggregating local descriptors into a compact image representation. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 3304–3311, 2010.

[20] G. Kim and A. Kim. Scan Context: Egocentric Spatial Descriptor for Place Recognition within 3D Point Cloud Map. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2018.

[21] X. Kong, X. Yang, G. Zhai, X. Zhao, X. Zeng, M. Wang, Y. Liu, W. Li, and F. Wen. Semantic Graph Based Place Recognition for 3D Point Clouds. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2020.

[22] L. Landrieu and M. Simonovsky. Large-scale Point Cloud Semantic Segmentation with Superpoint Graphs. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[23] J. Levinson and S. Thrun. Robust Vehicle Localization in Urban Environments Using Probabilistic Maps. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pages 4372–4378, 2010.

[24] 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 *Proc. of the IEEE Intl. Conf. on Computer Vision (ICCV)*, pages 2831–2840, 2019.

[25] I. Loshchilov and F. Hutter. Decoupled Weight Decay Regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[26] D. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *Intl. Journal of Computer Vision (IJCV)*, 60(2):91–110, 2004.

[27] S. Lowry and H. Andreasson. Lightweight, Viewpoint-Invariant Visual Place Recognition in Changing Environments. *IEEE Robotics and Automation Letters (RA-L)*, 3(2):957–964, 2018.

[28] W. Maddern, G. Pascoe, C. Linegar, and P. Newman. 1 year, 1000 km: The oxford robotcar dataset. *Intl. Journal of Robotics Research (IJRR)*, 36(1):3–15, 2017.

[29] T. Naseer, W. Burgard, and C. Stachniss. Robust Visual Localization Across Seasons. *IEEE Trans. on Robotics (TRO)*, 34(2):289–302, 2018.

[30] C.R. Qi, H. Su, K. Mo, and L.J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[31] C. Qi, K. Yi, H. Su, and L.J. Guibas. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *Proc. of the Advances in Neural Information Processing Systems (NIPS)*, 2017.

[32] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: an efficient alternative to sift or surf. In *Proc. of the IEEE Intl. Conf. on Computer Vision (ICCV)*, 2011.

[33] Ş. Săftescu, M. Gadd, D. De Martini, D. Barnes, and P. Newman. Kidnapped radar: Topological radar localisation using rotationally-invariant metric learning. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pages 4358–4364, 2020.

[34] M. Shakeri and H. Zhang. Illumination Invariant Representation of Natural Images for Visual Place Recognition. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2016.

[35] T. Shan, B. Englot, F. Duarte, C. Ratti, and D. Rus. Robust Place Recognition using an Imaging Lidar. *arXiv preprint*, 2021.

[36] J. Sivic and A. Zisserman. Video Google: A Text Retrieval Approach to Object Matching in Videos. In *Proc. of the IEEE Intl. Conf. on Computer Vision (ICCV)*, volume 2, pages 1470–1477, 2003.

[37] B. Steder, G. Grisetti, and W. Burgard. Robust Place Recognition for 3D Range Data Based on Point Features. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2010.

[38] B. Steder, M. Ruhnke, S. Grzonka, and W. Burgard. Place Recognition in 3D Scans Using a Combination of Bag of Words and Point Feature Based Relative Pose Estimation. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2011.

[39] B. Steder, R. Rusu, K. Konolige, and W. Burgard. NARF: 3D range image features for object recognition. In *Workshop on Defining and Solving Realistic Perception Problems in Personal Robotics at the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2010.

[40] H. Su, V. Jampani, D. Sun, S. Maji, E. Kalogerakis, M.H. Yang, and J. Kautz. SPLATNet: Sparse Lattice Networks for Point Cloud Processing. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[41] L. Sun, Z. Yan, A. Zaganidis, C. Zhao, and T. Duckett. Recurrent-OctoMap: Learning State-Based Map Refinement for Long-Term Semantic Mapping with 3D-Lidar Data. *IEEE Robotics and Automation Letters (RA-L)*, 2018.

[42] N. Sünderhauf, S. Shirazi, F. Dayoub, B. Upcroft, and M. Milford. On the performance of convnet features for place recognition. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 4297–4304, 2015.

[43] H. Thomas, C. Qi, J. Deschaud, B. Marcotegui, F. Goulette, and L. Guibas. KPConv: Flexible and Deformable Convolution for Point Clouds. In *Proc. of the IEEE Intl. Conf. on Computer Vision (ICCV)*, 2019.

[44] A. Uy and G. Lee. PointNetVLAD: Deep point cloud based retrieval for large-scale place recognition. In *Proc. of the IEEE Conf. on*

Computer Vision and Pattern Recognition (CVPR), pages 4470–4479, 2018.

[45] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Proc. of the Advances in Neural Information Processing Systems (NIPS)*, pages 5998–6008, 2017.

[46] I. Vizzo, X. Chen, N. Chebrolu, J. Behley, and C. Stachniss. Poisson Surface Reconstruction for LiDAR Odometry and Mapping. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2021.

[47] O. Vysotska and C. Stachniss. Lazy Data Association For Image Sequences Matching Under Substantial Appearance Changes. *IEEE Robotics and Automation Letters (RA-L)*, 1(1):213–220, 2016.

[48] O. Vysotska and C. Stachniss. Effective Visual Place Recognition Using Multi-Sequence Maps. *IEEE Robotics and Automation Letters (RA-L)*, 4(2):1730–1736, 2019.

[49] S. Wang, B.Z. Li, M. Khabsa, H. Fang, and H. Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020.

[50] X. Wei, I.A. Bârsan, S. Wang, J. Martinez, and R. Urtasun. Learning to Localize Through Compressed Binary Maps. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 10316–10324, 2019.

[51] L. Wiesmann, A. Milioto, X. Chen, C. Stachniss, and J. Behley. Deep Compression for Dense Point Cloud Maps. *IEEE Robotics and Automation Letters (RA-L)*, 6:2060–2067, 2021.

[52] Y. Xia, Y. Xu, S. Li, R. Wang, J. Du, D. Cremers, and U. Stilla. SOE-Net: A Self-Attention and Orientation Encoding Network for Point Cloud Based Place Recognition. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[53] S. Xie, J. Gu, D. Guo, C.R. Qi, L. Guibas, and O. Litany. Pointcontrast: Unsupervised pre-training for 3d point cloud understanding. In *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, pages 574–591, 2020.

[54] J. Yue-Hei Ng, F. Yang, and L.S. Davis. Exploiting local features from deep networks for image retrieval. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 53–61, 2015.

[55] W. Zhang and C. Xiao. PCAN: 3D attention map learning using contextual information for point cloud based retrieval. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 12436–12445, 2019.

[56] Z. Zhang, R. Girdhar, A. Joulin, and I. Misra. Self-supervised pretraining of 3d features on any point-cloud. *arXiv preprint arXiv:2101.02691*, 2021.

[57] H. Zhao, L. Jiang, J. Jia, P. Torr, and V. Koltun. Point transformer. *arXiv preprint arXiv:2012.09164*, 2020.