CAMBRIDGE
UNIVERSITY PRESS

RESEARCH ARTICLE

CrossMark

Condition-Invariant and Compact Visual Place Description by Convolutional Autoencoder

Hanjing Ye[#12], Weinan Chen[#3], Jingwen Yu[12], Li He[12], Yisheng Guan[3] and Hong Zhang[*12]

[1]Shenzhen Key Laboratory of Robotics and Computer Vision, Southern University of Science and Technology, Shenzhen, China.
[2]Department of Electronic and Electrical Engineering, Southern University of Science and Technology, Shenzhen, China.
[3]School of Mechanical and Electrical Engineering, Guangdong University of Technology, Guangzhou, China.
*Corresponding author. E-mail: hzhang@sustech.edu.cn.
#Hanjing Ye and Weinan Chen contribute equally to this paper.

Abstract

Visual place recognition (VPR) in condition-varying environments is still an open problem. Popular solutions are CNN-based image descriptors, which have been shown to outperform traditional image descriptors based on hand-crafted visual features. However, there are two drawbacks of current CNN-based descriptors: a) their high dimension and b) lack of generalization, leading to low efficiency and poor performance in real robotic applications. In this paper, we propose to use a convolutional autoencoder (CAE) to tackle this problem. We employ a high-level layer of a pre-trained CNN to generate features, and train a CAE to map the features to a low-dimensional space to improve the condition invariance property of the descriptor and reduce its dimension at the same time. We verify our method in four challenging real-world datasets involving significant illumination changes, and our method is shown to be superior to the state-of-the-art. The code of our work is publicly available in https://github.com/MedlarTea/CAE-VPR.

1. Introduction

Visual place recognition (VPR) is essential in autonomous robot navigation. VPR enables a robot to recognize previously visited places using visual data. VPR provides loop closure information for a SLAM algorithm to obtain a globally consistent map. Furthermore, VPR can support re-localization in a pre-built map of an environment. Due to its essential role, many VPR methods [23] have been proposed. However, in long-term navigation tasks, significant appearance variation, typically caused by seasonal change, illumination change, weather change and dynamic objects, such as those shown in Fig. 1, is still a challenge to VPR.

VPR is typically formulated as an image-matching procedure, which can be divided into two steps. The first step of VPR, also known as loop closure detection in the literature, selects candidates where map images are represented by global descriptors and a matching procedure between the map images and the current robot view can be performed in terms of image similarity. In the second step of VPR, verification is conducted via multi-view geometry, which uses keypoints in the images to determine if a query image (current robot view) is geometrically consistent with a candidate map image [22, 1, 33, 6, 9, 8, 40].

Figure 1. *To improve visual place recognition, we employ a CAE to compress a CNN-generated descriptor and gain a condition-invariant and low-dimensional image descriptor. This figure has shown the effectiveness of our descriptor. The top row is query images (current robot view), and the below row is matching images that are successfully matched by our descriptor. From left to right, they are from summer-winter traverse of Nordland [28], day-night scene of UACampus [21], and autumn-night and snow-night sequences with dynamics of RobotCar [24].*

In this paper, we focus on the first step of loop closure detection, namely, the generation of loop closure candidates efficiently and accurately. Traditionally, a global descriptor is obtained by aggregating the handcrafted local descriptors, like SIFT [22], ORB [33] and SURF [6]. In the case of significant appearance variations caused by, e.g., the day-night, season change and dynamic objects, handcrafted descriptors often fail to recognize places since locally keypoint descriptors can change significantly with the condition-dependent appearance. Convolutional neural networks (CNNs) have shown their advantages in various visual recognition tasks [20, 12, 32] and have been used to generate global image descriptors for visual loop closure detection. In [37], a pre-trained CNN is firstly used to produce a global descriptor directly. Alternatively, end-to-end trained descriptors with aggregating methods [2, 11, 30, 13] are proposed to gain higher performance.

However, deep learning-based VPR methods have some limitations. Firstly, a pre-trained CNN may generate descriptors easily with a dimension in the 10's of thousands, and hence result in time and storage problems. Secondly, the generalization ability of CNN descriptors is often poor. To tackle these limitations, we use a CAE to compress a CNN-generated descriptor and improve its generalization ability. Experiments on challenging datasets show that, by compressing the local feature maps of a CNN by CAE, the compressed descriptor achieves better results than the uncompressed descriptor in both seen and unseen environments at a lower computational cost.

2. RELATED WORK

2.1. *Visual Place Recognition*

In this paper, our concern is using a global descriptor to represent an image for loop closure detection. In the early works, VPR has been attained by extracting handcrafted local keypoints and descriptors firstly, such as SIFT [22], ORB [33] and SURF [6]. Then, these local features are aggregated to a global descriptor by vector quantization such as bag-of-words [36, 18, 29], VLAD [3] and Fisher Vectors [19]. Through clustering, a low-dimensional global descriptor can be achieved although spatial relations between the local descriptors are not encoded. Although these traditional methods have been widely used in SLAM (simultaneous localization and mapping) research, they still struggle in large-scale environments with severe appearance changes [23].

Recently, researchers have proposed to use CNNs to extract features for loop closure detection in large-scale environments. At first, pre-trained classification CNNs are directly used to extract dense local
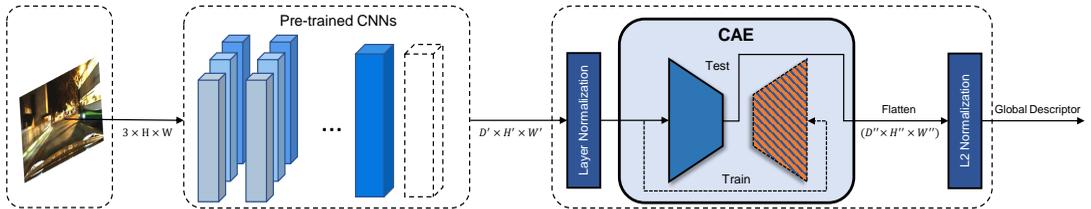
Figure 2. *The detailed pipeline of our system. Given an image with $3 \times H \times W$, CNNs extract the local feature map X_i with $D' \times H' \times W'$. The CNNs are classification pre-trained or VPR-trained, e.g. AlexNet, VGG16. Both are cut at the last convolutional layer (conv5), before ReLU. In the training time, CAE is trained unsupervised by a reconstruction loss. In the test time, the decoder part of CAE is not involved and the encoder part is kept to compress the normalized feature map and produce a low-dimensional global descriptor with $D'' \times H'' \times W''$. The global descriptor is then flattened and L2 normalized..*

feature maps [37, 34, 5], which serve as the visual features for visual place description. However, due to their high dimensions and inability to adapt to crowded environments, an end-to-end training model with a feature extractor and a pooling layer has been proposed, e.g., NetVLAD[2], SFRS[11], generalized-mean pooling [30], max pooling [38] and average pooling [31]. Although end-to-end models can perform well in crowded environments with low dimensions, training bias is introduced by training datasets. It leads to a poor generalization of the end-to-end trained descriptors to unseen environments. Here, we use the unsupervised method of CAE to learn an image descriptor by minimizing the reconstruction loss of the high-level features of a CNN instead. This enables the encoded descriptor to attain discriminative features and generalize to unseen environments with a lower dimension.

2.2. *Convolutional Autoencoder*

CAE has shown its superior performance in many applications. The first usage is to learn a feature extractor unsupervised by reconstructing input images as a pre-processing method, then finetuning the encoder with other downstream tasks[25]. Another usage is to learn a mapping function by reconstructing the input images to other domain information, e.g. flow image, depth image, path planning image and so on. CGAN (conditional generative adversarial nets) [27] and pix2pix [17] use CAE as their basic architecture to transfer, such as from day to night, from labels to faces and from edges to a photo. Moreover, in U-Net [32], semantic segmentation is achieved by a CAE-like architecture. Recently, Madhu *et al.* [39] propose a CAE-based GAN to estimate depth maps from night-time images. It is worth noting that it also uses the descriptor from the encoder to accomplish the day-night VPR task. Merrill *et al.* [26] utilize CAE to force the output of the decoder to be similar to the histogram of oriented gradients (HOG), and the output of the encoder is used as a global descriptor in the inference procedure.

Differently from the above usages of CAE, in this paper, we propose to use CAE to reconstruct the high-level features of the CNN, which is a post-processing method for decreasing the dimension of the features and promoting place recognition performance. The most similar to this idea is that Dai [10] uses a CAE to compress and fuse the local feature maps of the image patches for improving loop closure verification. Unlike this, the CAE in our method reconstructs the feature maps of the whole image, instead of feature maps of local image patches. In this way, our encoder can capture the most relevant features of the whole image for VPR.

3. APPROACH

In this section, we describe our network architecture and training strategy. The overall structure is shown in Fig. 2. In our framework, a local feature map is extracted from a pre-trained CNN. Specifically, the
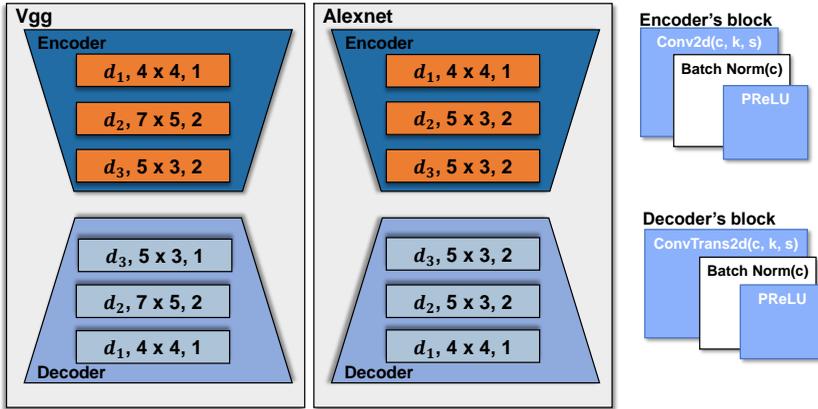
Figure 3. *The architecture of our CAE. Every encoder block contains Conv2d, Batch Normalization and PReLU layer. In the decoder block, Conv2d is replaced with ConvTranspose2d.* **c, k, s** *in the picture means* **channels, kernel size and stride**, *respectively.* d_3 *is a* **changeable parameter** *of the last convolutional kernel channels of the encoder, to produce the global descriptor with variable dimensions..*

local feature map is extracted by a high-level layer of a pre-trained CNN. The map is then normalized [4] and fed into the CAE. In the training procedure, the CAE consisting of an encoder and a decoder is trained by a reconstruction loss. However, in the inference step, the decoder part is dropped and only the encoder part is kept to produce the image descriptor.

3.1. Feature Extraction

Different layers of CNNs describe an image at different levels of semantics [37, 15]. In the VPR task, we choose the feature map of a deep layer, which is found in previous works to be condition-invariant and low-dimensional.

Similar to [2], we choose AlexNet [20] and VGG16 [35] as our backbone. The local feature map F is computed as:

$$F = f_\theta(I) \tag{1}$$

where I is an input image with a dimension of $3 \times H \times W$. H and W are the height and width of the input image. f_θ is a VPR-trained or pre-trained CNN without fine-tuning. In our work, F is from the last convolution layer of a CNN, before ReLU. For AlexNet, the dimension of F is $256 \times (\frac{1}{16}H - 2) \times (\frac{1}{16}W - 2)$. For VGG16, the dimension is $512 \times \frac{1}{16}H \times \frac{1}{16}W$. At such high dimensions, the global descriptor is of low computational efficiency to be stored and compared algorithmically for real-time performance. To tackle these problems, we use a CAE to compress the descriptor into a low-dimensional representation while promoting its condition-invariant capacity.

3.2. Convolutional Autoencoder

Given a high-dimensional local feature map F, we first normalize it with layer normalization [4]. Then, a CAE with three encoder layers and three decoder layers is trained to reconstruct the normalized feature

map. The architecture and the whole training strategy are:

$$\begin{aligned}
\hat{y} &= g_\theta(h(F)) \\
&= g_{dec}(g_{enc}(h(F))) \\
&\min L_{mse}(h(F), \ \hat{y})
\end{aligned} \tag{2}$$

where $h(x)$ is a layer normalization function [4], g_θ is a CAE with an encoder g_{enc} and a decoder g_{dec}. In the training procedure, we reconstruct the normalized local feature map $h(F)$ to train the CAE. We use mean squared error and backpropagation to reconstruct the normalized feature map $h(F)$. The mean squared error is defined as:

$$L_{mse} = \frac{1}{n} \|h(F) - \hat{y}\|_2^2 \tag{3}$$

where n is the dimension of the local feature map F. The layer normalization $h(x)$ is defined as:

$$h(x) = \frac{x - E[x]}{\sqrt{Var(x) + \epsilon}} \tag{4}$$

where x is a sample, $E[x]$ and $Var(x)$ are the mean and variance of the sample respectively, both of which are updated during training but frozen in the inference step. ϵ is a given value added to the denominator for numerical stability, which is set to 10^{-5} in our study. Empirically, a layer normalization can help to stable the optimization procedure and speed up the convergence.

Classic dimension reduction methods, e.g., PCA, only detect the linear relationship between features. For deep-learning-based pooling approaches, e.g., GeM [30], max pooling [38] and average pooling [31], they directly aggregate the $D' \times H' \times W'$ local feature map into a descriptor with D' dimensions. Because the features across spatial dimensions are directly aggregated, the spatial information in the feature map is therefore lost. In contrast, our CAE compresses the feature map non-linearly while maintaining the spatial relationship. In addition, the local feature map F is usually sparse and high-dimensional [7], indicating that only a few regions of a feature map have a solid response to a particular task like VPR or classification. With these attributes, our CAE can keep the most relevant features by reconstructing the input.

As shown in Fig. 3, in our CAE, each block in the encoder/decoder is composed of a convolutional/deconvolutional unit, a batch normalization unit [16] and a parametric rectified linear unit [14].

Since our CAE is based-on VGG16, the kernel sizes of three encoder blocks are 4×4, 7×5, 5×3, with strides 1, 2, 2, respectively. The channels of the first two encoder blocks are respectively d_1 and d_2. Similar to [10], to generate descriptors of different dimensions for comparison, the channels of the last encoder block d_3 are accordingly set to 8, 16, 32, 64, 128, 256 and 512. For AlexNet, the kernel sizes of three encoder blocks are 4×4, 5×3, 5×3, and the strides are 1, 2, 2, respectively. We adopt the same configurations as the encoder channels of VGG16 in our AlexNet encoder. For both architectures, the parameters of the decoder are similar to the encoder.

In the inference step, the decoder is not involved and the encoder is used to infer the compressed descriptor:

$$X = g_{enc}(h(F)) \tag{5}$$

where X is then flattened and L2-normalized to generate the final global descriptor. In the matching step, images are represented by the descriptors. Then a cosine similarity is utilized to find the best match in the reference set for a query image.

Table 1. *Summary of the experimental datasets.* **RobotCar** *(dbNight vs. qAutumn) indicates that a night sequence is used as the reference set (database) and an autumn sequence is the query set..*

Dataset	Environment	Traverse		Appearance Change
		Reference	Query	
Nordland	Train Journey	1415 (summer)	1415 (winter)	Very Strong
UACampus	Campus	647 (night)	647 (day)	Very Strong
RobotCar (dbNight vs. qAutumn)	Urban	7504 (night)	1046 (autumn)	Very Strong
RobotCar (dbNight vs. qSnow)	Urban	7504 (night)	1043 (snow)	Very Strong
RobotCar (dbSunCloud vs. qSnow)	Urban	7504 (sunCloud)	1043 (snow)	Strong
RobotCar (dbSunCloud vs. qAutumn)	Urban	7611 (sunCloud)	1046 (autumn)	Moderate

4. EXPERIMENTS SETUP

4.1. Dataset

To evaluate the performance of our proposed method, four datasets are utilized in the experiments, including Oxford RobotCar[24], Nordland[28] and UACampus[21] where only a part of the Oxford RobotCar is used as the training set and other datasets are used as the test set. These datasets contain significant appearance changes, in urban, train track, university campus and city simulation environments. Sample images from the datasets are shown in Fig. 1. A detailed description of the datasets is provided in Table 1 as well as below.

Oxford RobotCar [24] is an urban dataset that records a 10km route through central Oxford multiple times over one year. Within this dataset, challenging views with appearance changes are captured due to season, weather and time of the day. We choose a subset consisting of five sequences[1] involving sun cloud, autumn, snow and night environments, all of which contain strong appearance changes. To validate the effectiveness of our method, the dataset is separated with no overlap. Specifically, we extract a front-view image per meter for all sequences to construct the datasets. As shown in Fig. 4, the red route is the training set which includes 24k images, the green route is the test set, and the blue route represents the validation set. In the matching procedure, we have a query and a reference set. The query set contains the images of the green route, and the reference set includes the images of the whole route to increase the difficulty of matching. If the distance between a matched pair is within 25m, the decision is considered as a true positive.

Nordland [28] is a train journey dataset that contains significant seasonal changes. In this paper, summer and winter traverse are used as reference and query respectively. If the reference image is within two frames relative to the query, it is treated as a true positive.

UACampus [21] is a campus dataset with day-night illumination changes recorded on the campus of University of Alberta. Here, two subsets were captured in the morning (06:20) and evening (22:15) along the same route. Ground truth matching is available by manual annotation.

4.2. Evaluation Metric

Recall@1,5,10. To verify the overall performance of an image descriptor, we follow the common evaluation metric defined in [2], which is based on the top K nearest neighbors among all database descriptors to a query one. It can identify the matching ability of the descriptor in a tolerable interval. Matching is considered successful if the correct match exists within the top K nearest pairs. K is set to 1, 5 and 10 in our experiments.

---

[1]suncloud: 2014-12-09-13-21-02, night: 2014-12-10-18-10-50, 2014-12-16-18-44-24, autumn: 2014-11-18-13-20-12, snow: 2015-02-03-08-45-10
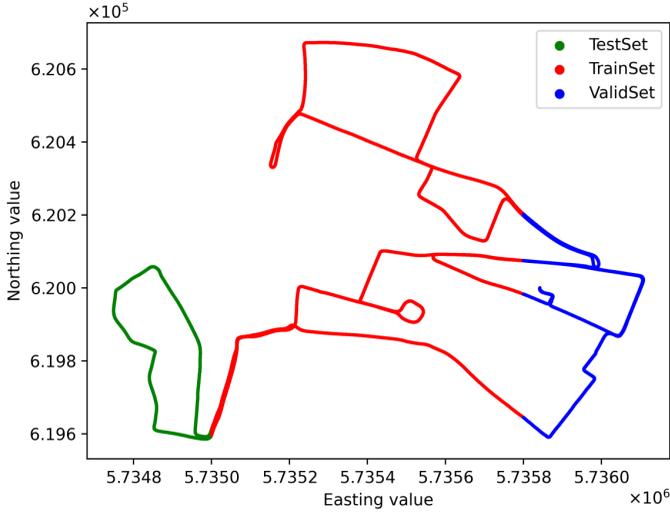
Figure 4. *Dataset separation for RobotCar which is strictly geometrically non-overlapped. The red route is for CAE training, the green route represents the test set, and the route with blue indicates the validation set..*

Precision-Recall Curve. Precision-Recall is another key evaluation metric in VPR. In the robotics field, top 1 matching pairs are vital because a decision must be made in the robot running. Given matched pairs and a threshold in terms of cosine similarity between image descriptors, we have the numbers of true positives, false positives and false negatives. precision and recall are defined as:

$$Precision = \frac{TP}{TP + FP}$$
$$Recall = \frac{TP}{TP + FN} \tag{6}$$

Multiple pairs of precision-recall (PR) values are produced by varying the threshold, and a PR-curve is regressed from the points formed by these pairs. A high threshold often causes low recall and high precision because a strict matching policy always reduces false positives (FP) but at the cost of many false negatives (FN). The ideal performance is when both precision and recall are high.

Average Precision (AP). The overall performance is usually represented by the average precision. It summarizes a precision-recall curve as the weighted mean of precisions achieved at all recall values:

$$AP = \sum_n (R_n - R_{n-1})P_n \tag{7}$$

where P_n and R_n are the precision and recall, respectively. n is the n_{th} threshold. Intuitively, AP is the integral of the precision-recall curve.

L2 distance distribution. To test the discriminative capacity of our global descriptor, we draw the histogram distribution of L2 distances of the true matches and the false matches. It can be used as prior knowledge to tell which distance interval can be trusted more for making a place recognition decision. Firstly, given a query vector, the reference vectors in the reference set within 25m (for RobotCar) are regarded as true matches and the remaining vectors are false matches. Secondly, L2 distance is calculated from a pair of vectors consisting of a query vector and a reference vector. Thirdly, traversing

the whole query set, we get L2 distances of all the true matched pairs and the false matched pairs. Lastly, L2 distances of the true matched pairs and false matches are formed as true matched and false matches histogram distributions respectively. Intuitively, a small overlap of two distributions indicates good discrimination.

4.3. Baseline and Our Method

NetVLAD [2] is a popular method trained on Pitts30k. We choose the best model by evaluating the method on the Pitts30k-valid. After training, we compress the descriptor dimension to 4096 with PCA and whitening. In our experiments, we use four tuples (one query, one positive, one negative) for training, for the purpose of reducing computational resource usage. **SFRS** [11] is the SOTA method which is also trained on Pitts30k with same network architecture as NetVLAD. The best model is obtained similarly as NetVLAD. **NetVLAD_VGG16** and **SFRS_VGG16** are the backbones of the Pitts30k-trained NetVLAD and SFRS rescpectively, and outputs the descriptors with $512 \times (\frac{1}{16}H) \times (\frac{1}{16}H)$ dimensions. **AlexNet** [20] is of the matconvnet version pre-trained in ImageNet, with the descriptor of $256 \times (\frac{1}{16}H - 2) \times (\frac{1}{16}H - 2)$ dimensions.
NetVLAD_VGG16+OursV is composed of **NetVLAD_VGG16** and the CAE introduced in Section 3.2. Similarly, **SFRS_VGG16+OursV** includes **SFRS_VGG16** and our CAE. **AlexNet+OursA** consists of **AlexNet** and our CAE.

Table 2. *Comparison of different settings of convolutional kernels of CAE in terms of R@1. d_1, d_2 and d_3 represent the number of channels of the three encoder blocks. c1 and c2 are different settings (kernel sizes and strides) of the encoder module where c1 would cause lower spatial dimension of the encoder output, c2 does the opposite. This experiment is conducted on RobotCar (dbNight vs. qSnow)..*

Method	d_1	d_2	d_3	c1	c2	Output Dimension	R@1
oursA	128	128	256	✓	✗	8192	0.739
	256	256	256	✓	✗	8192	0.742
	512	256	256	✓	✗	8192	0.755
	512	512	256	✓	✗	8192	0.734
	512	1024	256	✓	✗	8192	0.738
	256	256	12	✗	✓	8448	0.549
	512	256	12	✗	✓	8448	0.559
	512	512	12	✗	✓	8448	0.508
	512	1024	12	✗	✓	8448	0.527
oursV	128	128	256	✓	✗	8192	0.860
	256	256	256	✓	✗	8192	0.869
	512	256	256	✓	✗	8192	0.872
	512	512	256	✓	✗	8192	0.879
	512	1024	256	✓	✗	8192	0.879
	256	256	10	✗	✓	8160	0.806
	512	256	10	✗	✓	8160	0.804
	512	512	10	✗	✓	8160	0.809
	512	1024	10	✗	✓	8160	0.832

4.4. Implementation Details

In the experiments, the resolution of the input image is 640×480. For VGG16, the local feature map has a dimension of 614,400. For AlexNet, the dimension is 272,384. The hyperparameters of our CAE

are optimized empirically by experiments conducted in RobotCar (dbNight vs. qSnow). The results are shown in Table 2 where d_1, d_2 and d_3 represent the number of channels of the three encoder blocks, *c1* and *c2* imply the different settings (kernel sizes and strides) of the encoder blocks. Specifically, *c1* adopts the original setting as mentioned in Sec. 3.2 and *c2* indicates that the kernel size is 3×3 and the stride is 1. Here, c1 would cause lower spatial dimension of the encoder output, c2 does the opposite. For a balance of effectiveness and computation resource, d_1 and d_2 are set to 128, and *c1* is adopted.

During the CAE training, the backbone CNN is frozen. The Adam optimization algorithm is used to learn the model parameters, with a learning rate of 0.001 and a batch size of 128. The model is trained for 50 epochs. All the training is executed in PyTorch with 4 TITAN XP.

5. RESULTS AND DISCUSSION

5.1. *Effectiveness and Stability*

We first compare the performance of our method representative CNN-based image descriptors, namely, NetVLAD, SFRS and those from VGG16 and AlexNet. In Table 3, we can observe that NetVLAD and SFRS perform better on RobotCar than on Nordland and UACampus. VGG16 (the backbone of NetVlad and SFRS) shows quite different results in this test. On Nordland, SFRS_VGG16 surpasses SFRS by a significant margin with an AP of 0.969 versus 0.465 and recall@1 of 0.889 versus 0.282. Nevertheless, it is slightly worse with a recall@1 of 0.772 versus 0.834 on RobotCar (dbNight vs. qAutumn). This result could be attributed to the training bias introduced by the Pitts30k dataset, which is also an urban dataset similar to RobotCar. For VGG16, only conv5 and the following layer are trained. NetVLAD and SFRS, with VGG16 as their backbone, include a deep-learning-based VLAD module. Furthermore, the deep-learning-based VLAD module is optimized in the clustering space of the training datasets.

Compared to NetVLAD and NetVLAD_VGG16, OursV achieves better results with a higher AP, with the output dimension set to 4096 for a fair comparison. Even on a dataset with large appearance changes, such as RobotCar (dbNight vs. qSnow), the recall@1 of NetVLAD_VGG16+OursV is 0.861 versus NetVLAD's 0.691 and NetVLAD_VGG16's 0.523. It is worth noting that our method is unsupervised in this experiment on RobotCar, and it can nonetheless perform well in a non-urban dataset like Nordland and UACampus. To further validate the effectiveness and generalization ability of our method, we conduct experiments with different feature extractors, such as AlexNet pre-trained on ImageNet. AlexNet+OursA, which is also of dimension 4096, always produces better results than AlexNet, with an AP of 0.950 versus 0.657 in RobotCar (dbNight vs. qSnow) and recall@1 of 0.984 versus 0.956 in Nordland.

As shown in Table 3, our CAE is effective and memory-efficient on all datasets, and outperforms NetVLAD and SFRS and their backbones in most tests. Furthermore, at the dimension of 4096, the dimension of our descriptor is two orders of magnitude smaller than VGG16's 614,400 and AlexNet's 272,384.

5.2. *Comparison of Encoded Dimensions*

In this section, we will present the results from our study of the relationship between the output dimension of our CAE and matching performance. Fig. 5 shows the AP results in different datasets with the variation of the encoded dimensions. As shown in the left sub-figure, SFRS_VGG16+OursV and AlexNet+OursA achieve similar results to AlexNet. However, the output dimension of AlexNet is 272,384. Although the performance of both methods is a bit worse than AlexNet when the dimensions are small, e.g., 512 or 256, they still achieve better results than SFRS and SFRS_VGG16 with a moderate dimension.

From the right sub-figure, we can observe that, even in the urban-scale RobotCar dataset (dbNight vs. qSnow), OursV and OursA can achieve the same results as SFRS when the dimension is higher

Table 3. *Comparison of the baselines and our methods in terms of average precision (AP) and recall@1, 5, 10..*

| Dataset	Method	AP	R@1	R@5	R@10
Nordland	NetVLAD (4096d)	0.402	0.273	0.473	0.576
	NetVLAD_VGG16 (614,400d)	0.815	0.634	0.819	0.864
	NetVLAD_VGG16+OursV (4096d)	0.979	0.946	0.990	0.997
	SFRS (4096d)	0.465	0.282	0.522	0.630
	SFRS_VGG16 (614,400d)	0.969	0.889	0.970	0.988
	SFRS_VGG16+OursV (4096d)	0.979	0.946	0.990	0.997
	AlexNet (272,384d)	0.975	0.929	0.982	0.989
	AlexNet+OursA (4096d)	**0.984**	**0.962**	**0.996**	**0.999**
UACampus	NetVLAD (4096d)	0.744	0.674	0.788	0.838
	NetVLAD_VGG16 (614,400d)	0.996	0.934	0.971	0.986
	NetVLAD_VGG16+OursV (4096d)	**0.999**	0.969	0.992	0.995
	SFRS (4096d)	0.930	0.838	0.924	0.949
	SFRS_VGG16 (614,400d)	**0.999**	**0.985**	**0.995**	0.997
	SFRS_VGG16+OursV (4096d)	**0.999**	0.972	**0.995**	**0.998**
	AlexNet (272,384d)	0.993	0.932	0.978	0.985
	AlexNet+OursA (4096d)	**0.999**	0.977	0.994	0.995
RobotCar (dbNight vs. qAutumn)	NetVLAD (4096d)	0.933	0.759	0.874	0.914
	NetVLAD_VGG16 (614,400d)	0.865	0.644	0.719	0.754
	NetVLAD_VGG16+OursV (4096d)	**0.987**	**0.881**	0.913	0.928
	SFRS (4096d)	0.968	0.834	0.914	0.940
	SFRS_VGG16 (614,400d)	0.969	0.772	0.840	0.867
	SFRS_VGG16+OursV (4096d)	0.985	0.873	**0.918**	**0.932**
	AlexNet (272,384d)	0.916	0.741	0.801	0.845
	AlexNet+OursA (4096d)	0.952	0.832	0.884	0.903
RobotCar (dbNight vs. qSnow)	NetVLAD (4096d)	0.893	0.691	0.816	0.849
	NetVLAD_VGG16 (614,400d)	0.810	0.523	0.584	0.635
	NetVLAD_VGG16+OursV (4096d)	0.975	**0.861**	**0.891**	**0.907**
	SFRS (4096d)	0.943	0.726	0.831	0.874
	SFRS_VGG16 (614,400d)	0.944	0.738	0.814	0.849
	SFRS_VGG16+OursV (4096d)	**0.981**	0.836	**0.891**	**0.913**
	AlexNet (272,384d)	0.650	0.453	0.533	0.588
	AlexNet+OursA (4096d)	0.950	0.730	0.797	0.825
RobotCar (dbSunCloud vs. qSnow)	NetVLAD (4096d)	0.991	0.877	0.911	0.934
	NetVLAD_VGG16 (614,400d)	0.943	0.776	0.830	0.860
	NetVLAD_VGG16+OursV (4096d)	**0.995**	**0.919**	**0.941**	0.952
	SFRS (4096d)	0.992	0.903	0.939	**0.956**
	SFRS_VGG16 (614,400d)	0.990	0.889	0.923	0.942
	SFRS_VGG16+OursV (4096d)	0.993	0.910	0.936	0.948
	AlexNet (272,384d)	0.946	0.804	0.905	0.916
	AlexNet+OursA (4096d)	0.989	0.868	0.932	0.947
RobotCar (dbSunCloud vs. qAutumn)	NetVLAD (4096d)	0.996	0.928	0.956	0.965
	NetVLAD_VGG16 (614,400d)	0.972	0.820	0.879	0.906
	NetVLAD_VGG16+OursV (4096d)	0.996	0.930	0.962	0.971
	SFRS (4096d)	**0.999**	**0.958**	**0.979**	**0.984**
	SFRS_VGG16 (614,400d)	0.989	0.851	0.895	0.917
	SFRS_VGG16+OursV (4096d)	0.990	0.895	0.936	0.948
	AlexNet (272,384d)	0.991	0.899	0.932	0.948
	AlexNet+OursA (4096d)	0.992	0.909	0.931	0.945
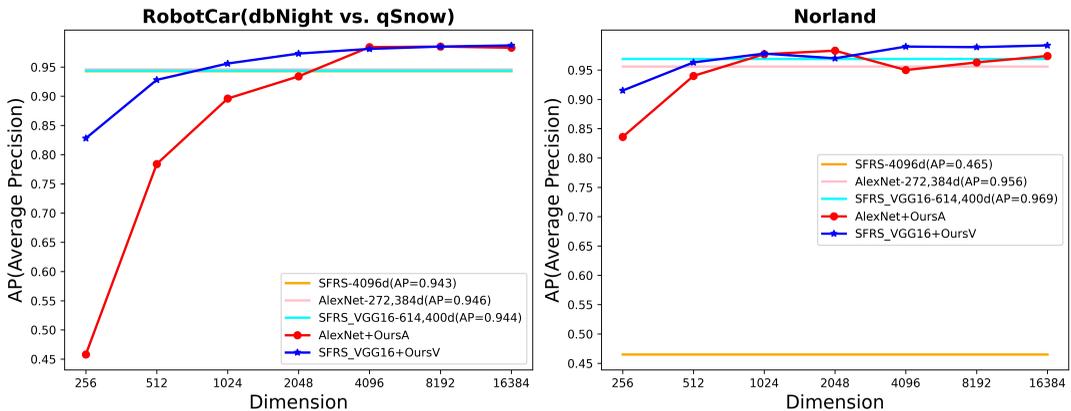
Figure 5. *Comparison of the baselines and our methods with different dimensions of our image descriptor in terms of AP on the Nordland and RobotCar (dbNight vs. qSnow).* **SFRS-4096d(AP=0.465)** *means that SFRS with 4096 dimensions output can achieve AP of 0.465, and others are similar indications. In the Nordland of the left picture, our method can achieve an AP of 0.97 with 1024d. In the RobotCar (dbNight vs. qSnow) of the right picture, SFRS_VGG16+OursV with just 1024d can approximately attain an AP of 0.96 as well as NetVLAD. While the dimension of SFRS is 4096..*

than 1024. From the above observation, we can infer that our CAE can attain high performance with low dimensions. However, as we continue to reduce the descriptor dimension, the performance will deteriorate.

5.3. Discriminative Capacity

We also plot the distribution of L2 distances between true matches and false matches, to evaluate the discriminating power of our CAE. For a fair comparison, we set the dimension of SFRS_VGG16+OursV as 4096, the same as SFRS. From Fig. 6(a), we can observe that the overlapping area of OursV is smaller than that of SFRS with a mean gap value of 0.322 versus 0.121. As shown in Fig. 6(b), the distributions of L2-distances between the true matches and false matches of SFRS are close where half of the true matches overlap with the false matches, resulting in a low mean gap value of 0.087. For SFRS_VGG16+OursV, the gap is 0.151, and half of the true matches do not overlap with the false ones.

These results show that our CAE is more discriminative than NetVLAD. However, the distributions of L2-distances between the true and false matches still overlap considerably. This could be caused by the fact that Nordland consists of only train road views, while RobotCar is more complicated with dynamic objects.

5.4. Ability of False Positives Avoidance

As mentioned in [23], false positive matches are fatal to VPR, since false matches lead to incorrect input to robot pose trajectory optimization. Consequently, recall at 100% precision is the prime metric for many tasks. From the result of the Nordland dataset shown in Fig. 7(a), SFRS_VGG16+OursV surpasses SFRS in terms of recall at 100% precision, while AlexNet+OursA performs poorly in this test. However, as shown in Fig. 7(b) of a RobotCar (dbNight vs. qSnow) experiment, AlexNet+OursA and SFRS_VGG16+OursV perform significantly better than other baselines.
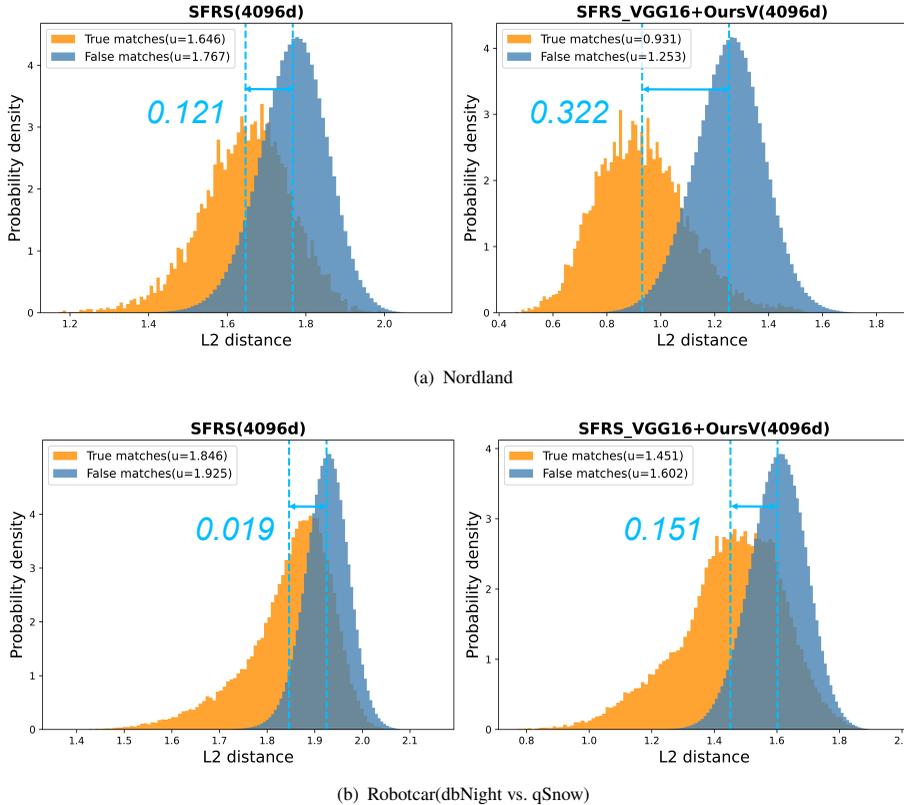
(a) Nordland

(b) Robotcar(dbNight vs. qSnow)

Figure 6. *L2 distribution of true and false matches for different methods. In both datasets, SFRS_VGG16+OursV (4096d) surpass SFRS a lot with difference of mean value of 0.322 versus 0.121 in Nordland and 0.151 versus 0.019 in RobotCar (dbNight vs. qSnow)..*

5.5. *Failure Cases Analysis*

Some true positive and false positive examples are shown in Fig. 8. From the left image pair, we can observe that a similar structure is a key to recognizing the same place. However, this might lead to a failure in the environments where similar structures widely exist, e.g., the environment shown in the middle image pair. In this wrongly matching pair, the tree distributions are similar, which is the reason for the recognition by our algorithm. However, these two places are not the same place. The right pair is also a failure case where dynamic objects occupy most of the image region. In this situation, our method would fail because not much discriminative information is captured. As the analysis of the above example, we conclude that loop closure verification is necessary for further accurate place recognition. In the verification, the local descriptors matching should have abilities of meaningless regions exclusion (e.g. dynamic objects in RobotCar) and adaptive attention on discriminative objects (e.g. trail direction in Nordland).

6. CONCLUSION

In this paper, we propose a simple method that uses a CAE in constructing an image descriptor from image feature maps from by a CNN. The experimental results have shown that the compressed CNN-descriptor by the CAE can attain high performance, better than state-of-the-art image descriptors such
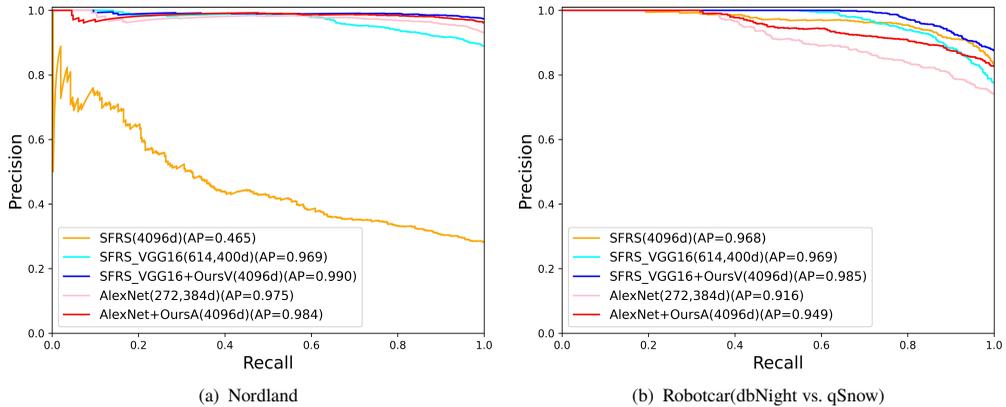
(a) Nordland

(b) Robotcar(dbNight vs. qSnow)

Figure 7. *Precision-Recall Curves for Nordland and Robotcar (dbNight vs. qSnow) datasets. The proposed method consistently outperforms better than the baselines with the metric of recall at 100% precision. In Nordland, OursV is the best with almost 0.2 recall at 100% precision. In Robotcar (dbNight vs. qSnow), SFRS_VGG16+OursV can attain nearly 0.7 recall at 100% precision..*
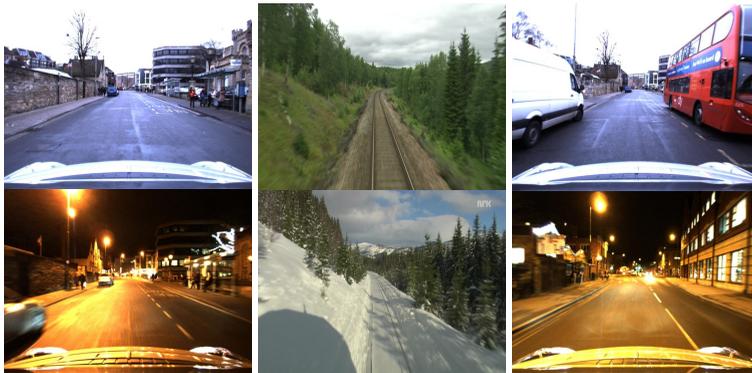

Figure 8. *The top row is query images and the below row is matching images. From left to right, they are true match, false match and false match..*

as NetVLAD, SFRS and than CNN-based descriptors at much higher dimensions such as VGG16 and AlexNet. Specifically, our CAE can consistently achieve a higher AP and recall than SFRS, when using the same descriptor dimension; in addition, our CAE achieves comparable results to other baseline descriptors when using a lower dimension than these descriptors. In RobotCar (dbNight vs. qSnow), OursV can achieve top-1 recall of 0.861 with 4096 dimensions, outperforming NetVLAD and SFRS. Furthermore, from the system perspective, our CAE can achieve higher recall at 100% precision than others. These quantitative results indicate that dimension reduction by our CAE can produce a compact and condition-invariant global descriptor while reducing the computational cost.

DECLARATION

- No Conflict of interest/Competing interests (check journal-specific guidelines for which heading to use)
- Ethics approval
- Consent to participate
- Consent for publication
- No availability of data and materials
- No code availability
- Hanjing Ye raised the main idea and completed the experiments and the draft. Weinan Chen helps with code work and idea revising. Jingwen Yu provided help with code work. Li He, Yisheng Guan and Hong Zhang shared their suggestions for revising the idea in this paper.

References

[1] A. Alahi, R. Ortiz, and P. Vandergheynst. Freak: Fast retina keypoint. In *2012 IEEE Conference on Computer Vision and Pattern Recognition* 2012, pp. 510–517. IEEE.

[2] 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* 2016, pp. 5297–5307.

[3] R. Arandjelovic and A. Zisserman. All about vlad. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition* 2013, pp. 1578–1585.

[4] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, (2016).

[5] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky. Neural codes for image retrieval. In *European conference on computer vision* 2014, pp. 584–599. Springer.

[6] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding,*, **110**(3), 346–359 (2008).

[7] Z. Chen, F. Maffra, I. Sa, and M. Chli. Only look once, mining distinctive landmarks from convnet for visual place recognition. in 2017 ieee. In *RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 9–16.

[8] J. Cheng, C. Wang, and M. Q.-H. Meng. Robust visual localization in dynamic environments based on sparse motion removal. *IEEE Transactions on Automation Science and Engineering,*, **17**(2), 658–669 (2019).

[9] J. Cheng, H. Zhang, and M. Q.-H. Meng. Improving visual localization accuracy in dynamic environments based on dynamic region removal. *IEEE Transactions on Automation Science and Engineering,*, **17**(3), 1585–1596 (2020).

[10] Z. Dai, X. Huang, W. Chen, C. Chen, L. He, S. Wen, and H. Zhang. Keypoint description by descriptor fusion using autoencoders. In *2020 IEEE International Conference on Robotics and Automation (ICRA)* 2020, pp. 65–71. IEEE.

[11] Y. Ge, H. Wang, F. Zhu, R. Zhao, and H. Li. Self-supervising fine-grained region similarities for large-scale image localization. In *European conference on computer vision* 2020, pp. 369–386. Springer.

[12] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* 2014, pp. 580–587.

[13] A. Gordo, J. Almazán, J. Revaud, and D. Larlus. Deep image retrieval: Learning global representations for image search. In *European conference on computer vision* 2016, pp. 241–257. Springer.

[14] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision* 2015, pp. 1026–1034.

[15] Y. Hou, H. Zhang, and S. Zhou. Convolutional neural network-based image representation for visual loop closure detection. In *2015 IEEE international conference on information and automation* 2015, pp. 2238–2245. IEEE.

[16] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning* 2015, pp. 448–456. PMLR.

[17] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* 2017, pp. 1125–1134.

[18] H. Jégou, M. Douze, C. Schmid, and P. Pérez. Aggregating local descriptors into a compact image representation. In *2010 IEEE computer society conference on computer vision and pattern recognition* 2010, pp. 3304–3311. IEEE.

[19] H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, and C. Schmid. Aggregating local image descriptors into compact codes. *IEEE transactions on pattern analysis and machine intelligence,*, **34**(9), 1704–1716 (2011).

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems,*, **25**, 1097–1105 (2012).

[21] Y. Liu, R. Feng, and H. Zhang. Keypoint matching by outlier pruning with consensus constraint. In *2015 IEEE International Conference on Robotics and Automation (ICRA)* 2015, pp. 5481–5486. IEEE.

[22] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision,*, **60**(2), 91–110 (2004).

[23] 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,*, **32**(1), 1–19 (2015).

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

[25] X.-J. Mao, C. Shen, and Y.-B. Yang. Image restoration using convolutional auto-encoders with symmetric skip connections. *arXiv preprint arXiv:1606.08921*, (2016).

[26] N. Merrill and G. Huang. Lightweight unsupervised deep loop closure. In *Proc. of Robotics: Science and Systems (RSS)* 2018, Pittsburgh, PA.

[27] M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, (2014).

[28] D. Olid, J. M. Fácil, and J. Civera. Single-view place recognition under seasonal changes. In *PPNIV Workshop at IROS 2018* 2018.

[29] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *2007 IEEE conference on computer vision and pattern recognition* 2007, pp. 1–8. IEEE.

[30] F. Radenović, G. Tolias, and O. Chum. Fine-tuning cnn image retrieval with no human annotation. *IEEE transactions on pattern analysis and machine intelligence,*, **41**(7), 1655–1668 (2018).

[31] A. S. Razavian, J. Sullivan, S. Carlsson, and A. Maki. Visual instance retrieval with deep convolutional networks. *ITE Transactions on Media Technology and Applications,*, **4**(3), 251–258 (2016).

[32] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention* 2015, pp. 234–241. Springer.

[33] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: An efficient alternative to sift or surf. In *2011 International conference on computer vision* 2011, pp. 2564–2571. IEEE.

[34] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops* 2014, pp. 806–813.

[35] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings* 2015.

[36] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *Computer Vision, IEEE International Conference on* 2003, volume 3, pp. 1470–1470. IEEE Computer Society.

[37] N. Sünderhauf, S. Shirazi, F. Dayoub, B. Upcroft, and M. Milford. On the performance of convnet features for place recognition. In *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)* 2015, pp. 4297–4304. IEEE.

[38] G. Tolias, R. Sicre, and H. Jégou. Particular object retrieval with integral max-pooling of cnn activations. In *ICLR 2016-International Conference on Learning Representations* 2016, pp. 1–12.

[39] M. Vankadari, S. Garg, A. Majumder, S. Kumar, and A. Behera. Unsupervised monocular depth estimation for night-time images using adversarial domain feature adaptation. In *European Conference on Computer Vision* 2020, pp. 443–459. Springer.

[40] Y.-T. Wang and G.-Y. Lin. Improvement of speeded-up robust features for robot visual simultaneous localization and mapping. *Robotica,*, **32**(4), 533–549 (2014).