

# Condition-Invariant and Compact Visual Place Description by Convolutional Autoencoder

Hanjing Ye<sup>1</sup>, Weinan Chen<sup>2</sup>, Jingwen Yu<sup>2</sup>, Li He<sup>2</sup>, Yisheng Guan<sup>1</sup> and Hong Zhang<sup>2\*</sup>

<sup>1</sup>Department of Mechanical and Electrical Engineering,  
Guangdong University of Technology, Guangzhou, 510006, China.

<sup>2</sup>Department of Electrical and Electronic Engineering, Southern University of Science and Technology, Shenzhen, 518055, China.

\*Corresponding author(s). E-mail(s): [hzhang@sustech.edu.cn](mailto:hzhang@sustech.edu.cn);

Contributing authors: [teamedlar@gmail.com](mailto:teamedlar@gmail.com);

[chenwn@sustech.edu.cn](mailto:chenwn@sustech.edu.cn); [11710414@mail.sustech.edu.cn](mailto:11710414@mail.sustech.edu.cn);

[hel@sustech.edu.cn](mailto:hel@sustech.edu.cn); [ysguan@gdut.edu.cn](mailto:ysguan@gdut.edu.cn);

## 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 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 three challenging 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>.

**Keywords:** Visual place recognition, Convolutional autoencoder, Unsupervised learning, Visual navigation



**Fig. 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 Norland [1], day-night scene of UACampus [2], and autumn-night and snow-night sequences with dynamics of RobotCar [3].

## 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 [4] 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 [5–8]. In this paper, we focus on the first step of loop closure detection, namely, generation of loop closure candidates efficiently and accurately. Traditionally, a global descriptor is obtained by aggregating the handcrafted local descriptors, like SIFT [5], ORB [7] and SURF [8]. 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 [9–11] and have been used to generate global image descriptors for visual loop closure detection. In [12], a pre-trained CNN is firstly used to produce a global descriptor directly. Alternatively, end-to-end trained descriptors with aggregating methods [13–15] 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 their ability to generalize. 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

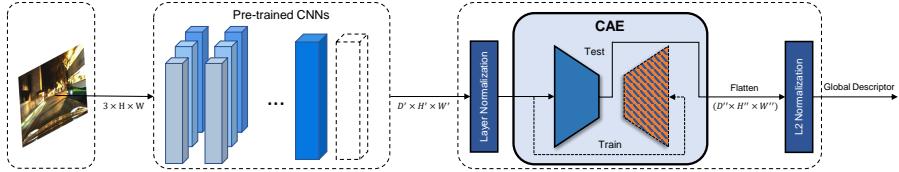
### 2.1 Visual Place Recognition

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 [5], ORB [7] and SURF [8]. Then, these local features are aggregated to a global descriptor by vector quantization such as bag-of-words [16–18], VLAD [19] and Fisher Vectors [20]. 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 [4].

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 feature maps [12, 21, 22], 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[13], generalized-mean pooling [14], max pooling [23] and average pooling [24]. 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. CGAN (conditional generative adversarial nets) [25] and pix2pix [26] use CAE as their basic architecture to encode features and generate images from input images in a source domain to that in a target domain, such as from day to night, from



**Fig. 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.

labels to facade and from edges to a photo. Moreover, in U-Net [11], semantic segmentation is achieved by a CAE-like architecture.

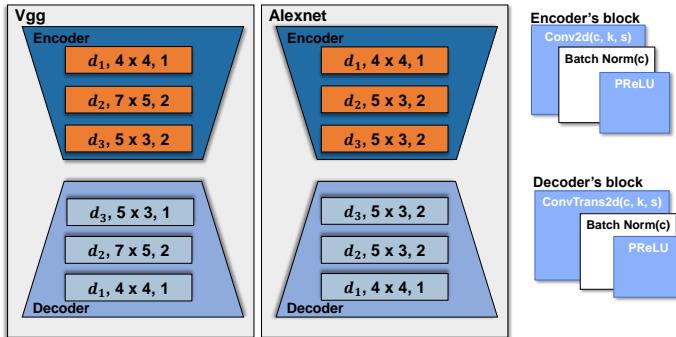
Recently, Madhu *et al.* [27] propose a CAE-based GAN to estimate depth maps from night-time images. It is worth noting that it also uses the descriptor from the encoder to accomplish the day-night VPR task. Merrill *et al.* [28] utilize CAE to force the output of the decoder to be similar to the histogram of oriented gradients (HOG), and the output of the encoder is used as a global descriptor in the inference procedure. Dai [29] uses a CAE to compress and fuse the local feature maps of the image patches for improving loop closure verification. Similar to Dai [29], our method trains a CAE to reconstruct the local feature maps of the CNNs and then uses the resulting encoder for generating image descriptors. Differently from [29], 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 local feature map is extracted by a high-level layer of a pre-trained CNN. The map is then normalized [30] 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 [12, 31]. 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.



**Fig. 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.

Similar to [13], we choose AlexNet [9] and VGG16 [32] as our backbone. The local feature map  $F$  is computed as:

$$F = f_\theta(I) \quad (1)$$

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.  $f_\theta$  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 firstly normalize it with layer normalization [30]. 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:

$$\begin{aligned} \hat{y} &= g_\theta(h(F)) \\ g_\theta &= [g_{enc} \ g_{dec}] \\ \min & (h(F), \hat{y}) \end{aligned} \quad (2)$$

where  $h(x)$  is a layer normalization function [30],  $g_\theta$  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

back propagation to reconstruct the normalized feature map  $h(F)$ . The mean squared error is defined as:

$$L_{mse} = \frac{1}{n} \|h(F) - g_\theta(h(F))\|_2^2 \quad (3)$$

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

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

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.  $\epsilon$  is a given value added to the denominator for numerical stability, which is set to  $10^{-5}$  in our study.

Classic dimension reduction methods, e.g., PCA, only detect the linear relationship between features. For deep-learning-based pooling approaches, e.g., GeM [14], max pooling [23] and average pooling [24], 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 [33], 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 [34] and a parametric rectified linear unit [35].

Since our CAE is based-on VGG16, the kernel sizes of three encoder blocks are  $4 \times 4$ ,  $7 \times 5$ ,  $5 \times 3$ , with strides 1, 2, 2, respectively. The channels of the first two encoder blocks are respectively  $d_1$  and  $d_2$ . Similar to [29], 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  $4 \times 4$ ,  $5 \times 3$ ,  $5 \times 3$ , 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:

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

where  $X$  is then flattened and L2-normalized to generate the final global descriptor.

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

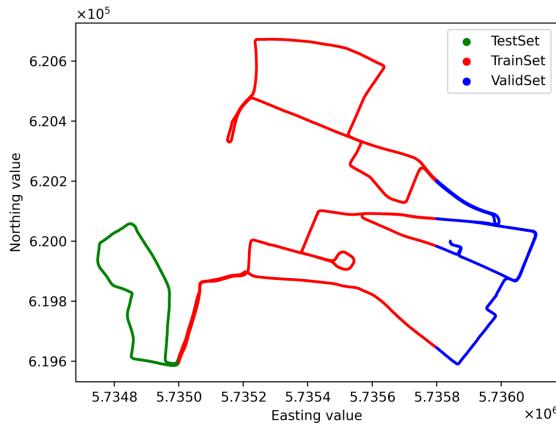
To evaluate the performance of our proposed method, we use three datasets in the experiments. These datasets contain significant appearance changes, in urban, train track and university campus environments. Sample images from the datasets are shown in Fig. 1. The detailed description of the datasets is provided in Table 1 as well as below.

**Oxford RobotCar** [3] 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<sup>1</sup> 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** [1] 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** [2] is a campus dataset with day-night illumination changes recorded on campus of University of Alberta. Here, two subsets were captured in the morning (06:20) and evening (22:15) along the same route. The ground truth matching is available by manual annotation.

<sup>1</sup>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



**Fig. 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.

## 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 [13], which is based on the top  $K$  nearest neighbors among all database descriptors to a query one. 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.

**Precision-Recall Curve.** Precision-Recall is another key evaluation metric in VPR. 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:

$$\begin{aligned} \text{Precision} &= \frac{TP}{TP + FP} \\ \text{Recall} &= \frac{TP}{TP + FN} \end{aligned} \quad (6)$$

Multiple pairs of precision-recall (PR) values are produced by varying the threshold, and a PR-curve can then be plotted for evaluation. 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:

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

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 distance of the true matches and the false matches. Their mean values are calculated to quantify the distinguishable ability of the descriptor. Intuitively, a small overlap of two distributions indicates good discrimination.

### 4.3 Baseline and Our Method

**NetVLAD** [13] 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 resources usage.

**VGG16** [32] is the backbone of the Pitts30k-trained NetVLAD, and outputs the descriptor with  $512 \times (\frac{1}{16}H) \times (\frac{1}{16}H)$  dimensions. **AlexNet** [9] 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. **OursV** is composed of **VGG16** and the CAE introduced in Section 3.2. **OursA** consists of **AlexNet** and our CAE.

**Table 2** Comparison of different settings of spatial dimensions and channel dimensions 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 output spatial dimensions of the encoder module. 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
	128	128	256	✓	✗	8192	0.860
oursV	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  $640 \times 480$ . 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 output spatial dimensions. Specifically,  $c1$  adopts the original settings and  $c2$  has a size of  $3 \times 3$  and the stride is 1. 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.

**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	0.402	0.273	0.473	0.576
	AlexNet	0.956	0.910	0.976	0.992
	VGG16	0.815	0.634	0.819	0.864
	OursA (4096d)	0.984	<b>0.962</b>	<b>0.996</b>	<b>0.999</b>
	OursV (4096d)	<b>0.979</b>	0.946	0.990	0.997
UACampus	NetVLAD	0.744	0.674	0.788	0.838
	AlexNet	0.993	0.932	0.974	0.985
	VGG16	0.996	0.934	0.971	0.986
	OursA (4096d)	<b>0.999</b>	<b>0.977</b>	<b>0.994</b>	<b>0.995</b>
	OursV (4096d)	0.999	0.969	0.992	<b>0.995</b>
RobotCar (dbNight vs. qAutumn)	NetVLAD	0.933	0.759	0.874	0.914
	AlexNet	0.950	0.827	0.879	0.906
	VGG16	0.865	0.644	0.719	0.754
	OursA (4096d)	0.952	0.832	0.884	0.903
	OursV (4096d)	<b>0.987</b>	<b>0.881</b>	<b>0.913</b>	<b>0.928</b>
RobotCar (dbNight vs. qSnow)	NetVLAD	0.893	0.691	0.816	0.849
	AlexNet	0.657	0.453	0.541	0.593
	VGG16	0.810	0.523	0.584	0.635
	OursA (4096d)	0.950	0.730	0.797	0.825
	OursV (4096d)	<b>0.975</b>	<b>0.861</b>	<b>0.891</b>	<b>0.907</b>
RobotCar (dbSunCloud vs. qSnow)	NetVLAD	0.991	0.877	0.911	0.934
	AlexNet	0.946	0.803	0.899	0.916
	VGG16	0.943	0.776	0.830	0.860
	OursA (4096d)	0.989	0.868	0.932	0.947
	OursV (4096d)	<b>0.995</b>	<b>0.919</b>	<b>0.941</b>	<b>0.952</b>
RobotCar (dbSunCloud vs. qAutumn)	NetVLAD	0.996	0.928	0.956	0.965
	AlexNet	0.991	0.902	0.932	0.942
	VGG16	0.972	0.820	0.879	0.906
	OursA (4096d)	0.992	0.909	0.931	0.945
	OursV (4096d)	<b>0.996</b>	<b>0.930</b>	<b>0.962</b>	<b>0.971</b>

## 5 RESULTS AND DISCUSSION

### 5.1 Effectiveness and Stability

We first compare the performance of our method representative CNN-based image descriptors, namely, NetVLAD and those from VGG16 and AlexNet. In Table 3, we can observe that NetVLAD performs better on RobotCar than on Norland and UACampus. VGG16 (the backbone of NetVlad) shows quite different results in this test. On Norland, VGG16 surpasses NetVLAD by a significant margin with an AP of 0.815 versus 0.402 and recall@1 of 0.634 versus 0.273. Nevertheless, it is slightly worse with an AP of 0.865 versus 0.933 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, with VGG16 as its backbone, includes a deep-learning-based VLAD module. Furthermore, the deep-learning-based VLAD module is optimized in the clustering space of the training datasets.

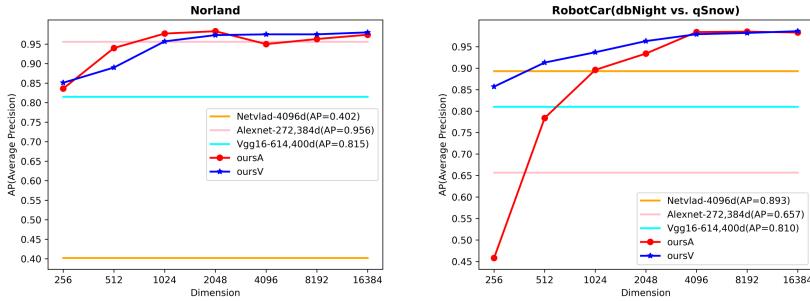
Compared to NetVLAD and VGG16, OursV achieves better results with a higher AP, with the output dimension set to 4096 for a fair comparison. Even on a dataset with large appearance changes, such as RobotCar (dbNight vs. qSnow), the recall@1 of OursV is 0.861 versus NetVLAD's 0.691 and VGG16's 0.523. It is worth noting that our method is unsupervised in this experiment on RobotCar, and it can nonetheless perform well in the non-urban dataset like Norland 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. OursA, which is also of dimension 4096, always produces better results than AlexNet, with an AP of 0.975 versus 0.657 in RobotCar (dbNight vs. qSnow) and recall@1 of 0.962 versus 0.910 in Norland.

As shown in Table 3, our CAE is effective and memory-efficient on all datasets, and outperforms NetVLAD 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, OursV and 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 NetVLAD and 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



**Fig. 5** Comparison of the baselines and our methods with different dimensions of our image descriptor in terms of AP on the Norland and RobotCar (dbNight vs. qSnow). **NetVLAD-4096d(AP=0.402)** means that NetVLAD with 4096 dimensions output can achieve AP of 0.402, and others are similar indications. In the Norland of the left picture, our method can achieve an AP of 0.95 with 1024d. In the RobotCar (dbNight vs. qSnow) of the right picture, OursV with just 1024d can approximately attain an AP of 0.90 as well as NetVLAD. While the dimension of NetVLAD is 4096.

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

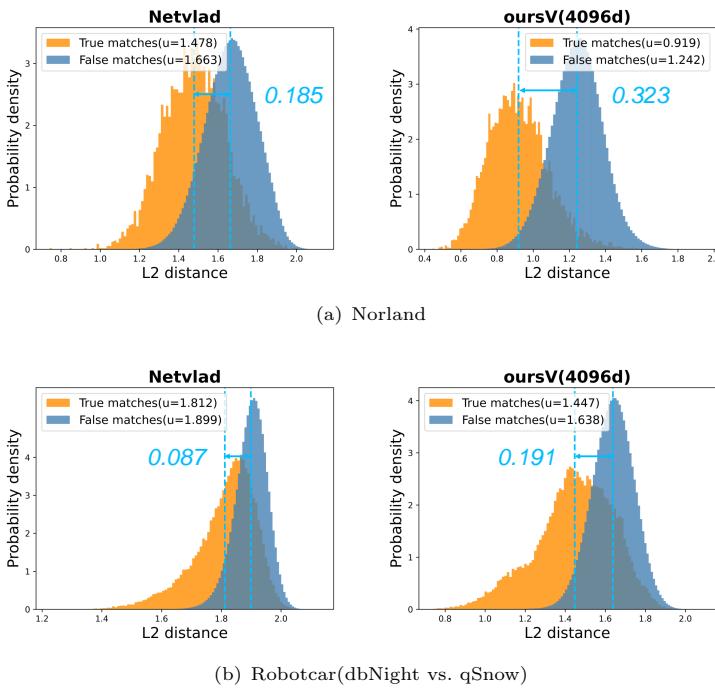
### 5.3 Discriminative Capacity

We also plot the distribution of L2-distances between true matches, false matches, to evaluate the discriminating power of our CAE. For a fair comparison, we set the dimension of OursV as 4096, the same as NetVLAD. From Fig. 6(a), we can observe that the overlapping area of OursV is smaller than that of NetVLAD with a mean gap value of 0.323 versus 0.185. As shown in Fig. 6(b), the distributions of L2-distances between the true matches and false matches of NetVLAD are close where half of the true matches overlap with the false matches, resulting in a low mean gap value of 0.087. For OursV, the gap is 0.191, 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 Norland consists of only train road views, while RobotCar is more complicated with dynamic objects.

### 5.4 Ability of False Positives Avoidance

As mentioned in [4], 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 Norland dataset shown in Fig. 7(a), OursV surpasses VGG16 and AlexNet in terms of recall at 100% precision, while OursA and NetVLAD perform

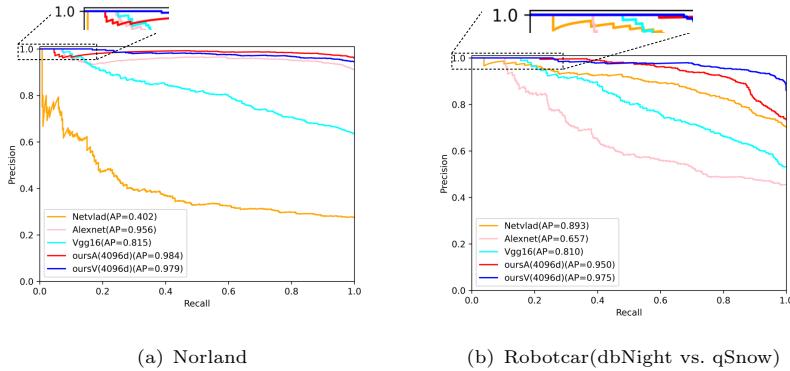


**Fig. 6** L2 distribution of true and false matches for different methods. In both datasets, OursV (4096d) surpass NetVLAD a lot with difference of mean value of 0.323 versus 0.185 in Norland and 0.191 versus 0.087 in RobotCar (dbNight vs. qSnow).

poorly in this test. However, as shown in Fig. 7(b) of a RobotCar (dbNight vs. qSnow) experiment, OursA and OursV perform significantly better than other baselines.

## 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 as NetVLAD 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 NetVLAD, 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. 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.



**Fig. 7** Precision-Recall Curves for Norland and Robotcar (dbNight vs. qSnow) datasets. The proposed method consistently outperforms better than the baselines with the metric of recall at 100% precision. In Norland, OursV is the best with almost 0.2 recall at 100% precision. In Robotcar (dbNight vs. qSnow), OursV and OursA attain nearly 0.3 recall at 100% precision.

## DECLARATION

- This work was supported in part by the Leading Talents Program of Guangdong Province under Grant No. 2016LJ06G498 and 2019QN01X761
- 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 and 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] Olid, D., Fácil, J.M., Civera, J.: Single-view place recognition under seasonal changes. arXiv preprint arXiv:1808.06516 (2018)
- [2] Liu, Y., Feng, R., Zhang, H.: Keypoint matching by outlier pruning with consensus constraint. In: 2015 IEEE International Conference on Robotics and Automation (ICRA), pp. 5481–5486 (2015). IEEE
- [3] Maddern, W., Pascoe, G., Linegar, C., Newman, P.: 1 year, 1000 km: The oxford robotcar dataset. The International Journal of Robotics Research **36**(1), 3–15 (2017)

- [4] Lowry, S., Sünderhauf, N., Newman, P., Leonard, J.J., Cox, D., Corke, P., Milford, M.J.: Visual place recognition: A survey. *IEEE Transactions on Robotics* **32**(1), 1–19 (2015)
- [5] Lowe, D.G.: Distinctive image features from scale-invariant keypoints. *International journal of computer vision* **60**(2), 91–110 (2004)
- [6] Alahi, A., Ortiz, R., Vandergheynst, P.: Freak: Fast retina keypoint. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition, pp. 510–517 (2012). Ieee
- [7] Rublee, E., Rabaud, V., Konolige, K., Bradski, G.: Orb: An efficient alternative to sift or surf. In: 2011 International Conference on Computer Vision, pp. 2564–2571 (2011). Ieee
- [8] Bay, H., Ess, A., Tuytelaars, T., Van Gool, L.: Speeded-up robust features (surf). *Computer vision and image understanding* **110**(3), 346–359 (2008)
- [9] Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems* **25**, 1097–1105 (2012)
- [10] Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 580–587 (2014)
- [11] Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical Image Computing and Computer-assisted Intervention, pp. 234–241 (2015). Springer
- [12] Sünderhauf, N., Shirazi, S., Dayoub, F., Upcroft, B., Milford, M.: On the performance of convnet features for place recognition. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 4297–4304 (2015). IEEE
- [13] Arandjelovic, R., Gronat, P., Torii, A., Pajdla, T., Sivic, J.: Netvlad: Cnn architecture for weakly supervised place recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5297–5307 (2016)
- [14] Radenović, F., Tolias, G., Chum, O.: Fine-tuning cnn image retrieval with no human annotation. *IEEE transactions on pattern analysis and machine intelligence* **41**(7), 1655–1668 (2018)
- [15] Gordo, A., Almazán, J., Revaud, J., Larlus, D.: Deep image retrieval:

- Learning global representations for image search. In: European Conference on Computer Vision, pp. 241–257 (2016). Springer
- [16] Sivic, J., Zisserman, A.: Video google: A text retrieval approach to object matching in videos. In: Computer Vision, IEEE International Conference On, vol. 3, pp. 1470–1470 (2003). IEEE Computer Society
  - [17] Jégou, H., Douze, M., Schmid, C., Pérez, P.: Aggregating local descriptors into a compact image representation. In: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 3304–3311 (2010). IEEE
  - [18] Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Object retrieval with large vocabularies and fast spatial matching. In: 2007 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8 (2007). IEEE
  - [19] Arandjelovic, R., Zisserman, A.: All about vlad. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1578–1585 (2013)
  - [20] Jégou, H., Perronnin, F., Douze, M., Sánchez, J., Pérez, P., Schmid, C.: Aggregating local image descriptors into compact codes. IEEE transactions on pattern analysis and machine intelligence **34**(9), 1704–1716 (2011)
  - [21] Sharif Razavian, A., Azizpour, H., Sullivan, J., Carlsson, S.: Cnn features off-the-shelf: an astounding baseline for recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 806–813 (2014)
  - [22] Babenko, A., Slesarev, A., Chigorin, A., Lempitsky, V.: Neural codes for image retrieval. In: European Conference on Computer Vision, pp. 584–599 (2014). Springer
  - [23] Tolias, G., Sicre, R., Jégou, H.: Particular object retrieval with integral max-pooling of cnn activations. arXiv preprint arXiv:1511.05879 (2015)
  - [24] Razavian, A.S., Sullivan, J., Carlsson, S., Maki, A.: Visual instance retrieval with deep convolutional networks. ITE Transactions on Media Technology and Applications **4**(3), 251–258 (2016)
  - [25] Mirza, M., Osindero, S.: Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784 (2014)
  - [26] Isola, P., Zhu, J.-Y., Zhou, T., Efros, A.A.: Image-to-image translation

- with conditional adversarial networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1125–1134 (2017)
- [27] Vankadari, M., Garg, S., Majumder, A., Kumar, S., Behera, A.: Unsupervised monocular depth estimation for night-time images using adversarial domain feature adaptation. In: European Conference on Computer Vision, pp. 443–459 (2020). Springer
- [28] Merrill, N., Huang, G.: Lightweight unsupervised deep loop closure. arXiv preprint arXiv:1805.07703 (2018)
- [29] Dai, Z., Huang, X., Chen, W., Chen, C., He, L., Wen, S., Zhang, H.: Keypoint description by descriptor fusion using autoencoders. In: 2020 IEEE International Conference on Robotics and Automation (ICRA), pp. 65–71 (2020). IEEE
- [30] Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization. arXiv preprint arXiv:1607.06450 (2016)
- [31] Hou, Y., Zhang, H., Zhou, S.: Convolutional neural network-based image representation for visual loop closure detection. In: 2015 IEEE International Conference on Information and Automation, pp. 2238–2245 (2015). IEEE
- [32] Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
- [33] Chen, Z., Maffra, F., Sa, I., Chli, M.: 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
- [34] Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International Conference on Machine Learning, pp. 448–456 (2015). PMLR
- [35] He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1026–1034 (2015)