Explicit feature disentanglement for visual place recognition across appearance changes

Li Tang¹, Yue Wang¹, Qimeng Tan² and Rong Xiong¹

Abstract
In the long-term deployment of mobile robots, changing appearance brings challenges for localization. When a robot travels to the same place or restarts from an existing map, global localization is needed, where place recognition provides coarse position information. For visual sensors, changing appearances such as the transition from day to night and seasonal variation can reduce the performance of a visual place recognition system. To address this problem, we propose to learn domain-unrelated features across extreme changing appearance, where a domain denotes a specific appearance condition, such as a season or a kind of weather. We use an adversarial network with two discriminators to disentangle domain-related features and domain-unrelated features from images, and the domain-unrelated features are used as descriptors in place recognition. Provided images from different domains, our network is trained in a self-supervised manner which does not require correspondences between these domains. Besides, our feature extractors are shared among all domains, making it possible to contain more appearance without increasing model complexity. Qualitative and quantitative results on two toy cases are presented to show that our network can disentangle domain-related and domain-unrelated features from given data. Experiments on three public datasets and one proposed dataset for visual place recognition are conducted to illustrate the performance of our method compared with several typical algorithms. Besides, an ablation study is designed to validate the effectiveness of the introduced discriminators in our network. Additionally, we use a four-domain dataset to verify that the network can extend to multiple domains with one model while achieving similar performance.

Keywords
Place recognition, feature disentanglement, adversarial, self-supervised, changing appearance

Date received: 07 December 2020; accepted: 25 June 2021

Introduction
Place recognition is a vital ability for robots. Ranging from autonomous driving to flying robots for precision agriculture, different kinds of sensors are leveraged in different localization scenarios, among which cameras are gaining more and more popularity. The main advantage of imagery sensors is their low cost, compared to expensive light detection and ranging (LiDAR), inertial navigation system (INS), etc. Visual localization has been studied for years, and many visual simultaneous localization and mapping
(SLAM) systems are proposed, which achieve impressive performance under ideal conditions. Visual place recognition plays two roles in a SLAM system: (1) in the mapping stage, robots need to find loop closure, so as to reduce drifting and build a global consistency map and (ii) in the localization stage, localization may sometimes fail, such as the kidnapped robot problem, which also needs loop detection. Most loop detection algorithms firstly try to coarsely relocalize the robots against a known database using place recognition, followed by pose estimation. Thus, visual place recognition may affect mapping accuracy and relocalization success rate.

As the community pays more and more attention to scenes with appearance changes, such as the urban environment, challenges for visual place recognition also come. A typical visual place recognition pipeline includes feature extraction, feature matching, and temporal fusion, among which feature extraction is now a bottleneck. The recent success of deep learning has shown a great potential for a neural network to become a robust feature extractor. Thus, this article tries to improve the feature extraction module using deep learned features. The main challenge for handcrafted visual features is that they are sensitive to appearance changes, such as a shift from day to night and seasonal transitions. Some methods exploit deep learned features from supervised learning. When testing data are similar to training data, these supervised methods perform very well. However, they require massive manually labeled data, which are labor-intensive and time-consuming. That may be unsuitable for some place recognition scenarios. Thus, self-supervised or unsupervised methods are preferred. Autoencoder learns features in a self-supervised way, and the output of the encoder is shown to be suitable for place recognition. In their work, training data and testing data are quite different; therefore, it pays more attention to generalization ability. Compared to supervised learning, it sacrifices performance to obtain generalization. To find a good balance between performance and generalization, some researchers assume that training and testing data can share some similar properties, such as the same place, the same sensors, and similar illumination, but without position-level alignment. This assumption is reasonable in some applications, such as inspection robots working in the same place at different times. ToDayGAN translated from nighttime images to daytime style using generative adversarial network (GAN), and the generated images are used to match with images captured in the daytime. Although nighttime images in the training and testing phase look similar, they still have subtle appearance differences because they are captured at different times. Our article is also targeted at this setting. The main difference is that features in ToDayGAN are extracted from the translated images after the image translation, while our method directly puts constraints on features.

In this article, we are set to construct the neural network that can explicitly disentangle domain-unrelated and domain-related features of images from different domains, and the domain-unrelated features are used for place recognition. The only supervised information is the domain, where a domain denotes a specific appearance condition, such as daytime in spring or nighttime in summer. For example, by assuming that the appearance does not change too much during a short period, we can label all the images as some domain according to the appearance condition at that time. By grouping images into different domains, the proposed network can find the invariant information among them. This idea is motivated by the hypothesis that in the place recognition application, an image can be seen as the composition of place content and appearance content, where the place is the domain-unrelated content of a scene (e.g. corners or edges of buildings), whereas appearance is the domain-related properties (e.g. brightness of sunlight and type of season). Under this hypothesis, based on the definition of disentangled representation, we disentangle place and appearance features using two encoders, which are part of an autoencoder. To ensure that the appearance feature only corresponds to domain-related content, an adversarial loss is applied on pairs of place features and appearance features. Besides, another adversarial loss is designed to constrain the place features mapping to the same latent space, such that the appearance feature only corresponds to domain-related information. As a result, the place feature is robust against appearance changes, and it is used as the descriptor in visual place recognition. The network is trained in a self-supervised manner without the requirement for aligned image sequences, and only domain information is needed, which is also called weak-supervised in some articles. This can be easily achieved by firstly collecting sessions of images under the same conditions and then marking each session as one domain. Besides, the network is shared across different domains, which allows our model to be adaptable to more domains without increasing the number of parameters. The main contributions of this article are listed as follows:

- A data modeling method is presented for visual place recognition, and a self-supervised feature learning method is proposed to disentangle the domain-unrelated and domain-related content from multiple-domain images.
- A disentangled feature learning network based on adversarial learning is proposed, which is able to be extended to multiples domains without increasing model complexity. This makes our network feasible for applications with limited resources.
- Two toy case studies are carried out to validate our feature disentanglement method with qualitative and quantitative results. We also try to interpret our network in this part.
- Experiments for place recognition are conducted on three public dataset and one newly proposed dataset. Our method shows favorable performance. We also
open the source code for reproduction (https://github.com/dawnos/fdn-pr).

This article is an extensive study based on our previous work. One additional contribution is that we improve our network to get higher performance by reconstructing high-level image feature instead of the original image. Another one is that we try to interpret our network through theoretical discussion and ablation study. Finally, more datasets and comparison methods are employed to verify the proposed disentanglement method.

The remainder of this article is organized as follows: related work is discussed and summarized in the second section. Then in the third section, we present the data model used in this article. Our method will be presented thoroughly in the fourth section. We will introduce the experiments in detail and show results in the fifth section. The conclusion will be made in the sixth section.

Related work
Typical pipeline for visual place recognition includes (1) feature extraction and (2) matching optionally followed by (3) temporal fusion. This article will focus on the feature extraction module.

Handcraft features
In the early years, handcrafted features are used for place recognition. These features can be classified into two categories, namely global features and local features. Methods based on handcrafted global features try to assign appearance-invariant features to each image directly. In some early works, histograms of oriented gradient (HOG) features are used as descriptors to compute the distance between database and query image, where the gradient is able to overcome simple illumination changes. Later, the GIST descriptor, which represents the high-frequency part of an image, is also leveraged because the human eye is more sensitive to it. On the other hand, local-feature-based methods firstly extract a bunch of local features and then aggregate them into global features. In place recognition, performance is limited by the choice of local features, thus local features with an appearance-invariant property are preferred. Gradient-based local features such as Scale-Invariant Feature Transform (SIFT) and Oriented FAST and Rotated BRIEF (ORB) are found to be robust to small appearance changes. Especially, the ORB descriptor is used in the place recognition (loop closure) module of the widely used ORB-SLAM system. In addition, different aggregating algorithms are designed to generate global descriptors, such as a bag of visual words and Vector of Locally Aggregated Descriptor (VLAD).

Due to the limited robustness of handcrafted features, these methods are sensitive to extreme appearance changes, thus not preferable in visual place recognition.

Supervised features
After the great success of deep neural networks in the computer vision area in recent years, researchers start to explore how can visual place recognition benefit from deep learning. In one of the earliest trials, different layers in AlexNet are reported to have different place recognition performances. The authors find that features from the middle layers are more robust against changing appearance. However, the AlexNet features are not good enough, because the network is pretrained on the imagenet large-scale visual recognition challenge dataset, which is different from the testing set. To go further, different supervised methods are proposed, which constrain the extracted features of images from the same place to be similar. One way is to use a classification network for place recognition where each place is a class, and this method can achieve comparable results. The network is trained and tested on a dataset captured from static cameras at different times. This work does not need manual labeling, and it shows the potential for the neural network to distinguish places. However, as places are determined once training is finished, it cannot be extended to other scenes.

To make the network more flexible, NetVLAD improves the competitive aggregating method VLAD by using soft assignment, which becomes a differential module. With the module, the feature network can be optimized by triplet loss. Labeled data from Google Street View Time Machine are needed to construct training tuple. Besides, another research assigns multiple images to one place and fuses their features together to boost performance, which uses two datasets aligned by Global Positioning System (GPS). In addition to improving the features, finding useful regions for place recognition can also help, such as stable return on investments and attention maps. These supervised methods achieve impressive results, but the requirement for massive labeled data may be hard to fulfill in the fast deployment.

It is also possible to make the extracted features more robust by postprocessing before the feature matching module. For example, the original AlexNet features can be further improved by applying PCA to those features followed by only keeping the components with small eigenvalues, because the principal components with large eigenvalues represent variations in the images. Additionally, quantifying the features using hashing can speed up the matching process. These methods are not competitors of our method, and instead, they can be added to any feature extractors to get high performance.
Self-supervised features

Another branch of machine learning methods, namely self-supervised learning, does not rely on aligned data. Autoencoder\textsuperscript{44} is a popular self-supervised network architecture, which is used in many place recognition researches.\textsuperscript{11,45–47} The output features of encoders exhibit robustness in place recognition. The robustness of the learned features can be improved by reconstructing corrupted input images,\textsuperscript{47} or trying to reconstruct HOG of the input images.\textsuperscript{11} One advantage of the latter one is that their training sets are different from the testing set, demonstrating favorable generalization.

In recent years, adversarial learning is getting more and more attention in the literature on self-supervised learning. Inspired by the work of style transfer\textsuperscript{13,48,49} using GAN,\textsuperscript{50} some researchers try to transfer query images to match the style of database images, followed by local feature matching, global descriptor matching, or dense matching.\textsuperscript{12,51–54} For example, ToDayGAN transfers nighttime images into daytime ones, extracts features using DenseVLAD,\textsuperscript{55} and finally matches with daytime images in the database.\textsuperscript{12} Each model in these methods is targeted at the two domains used in the training phase. When adding new domains, new models are needed, and the number of models increases rapidly. Another method is enhancing the pretrained NetVLAD\textsuperscript{10} with semantic information, which is shown to be viewpoint-invariant.\textsuperscript{56} It demonstrates that appearance-based descriptors, such as our method, can be improved to overcome changing viewpoints with this technique.

Data modeling

In this section, we formulate our problem and define disentangled representation. The input data are modeled as a multiple-domain generative process, and the feature extraction module is modeled as an inference process. These two processes are summarized in Figure 1. Based on these, we derive our definition at the end of this section.

Generative process

Images are the reflections of the world. Let $\mathcal{W}$ be the state space of the world. An image $x \in \mathcal{X}$ can be seen as the observation of a state in $\mathcal{W}$, where $\mathcal{X}$ is the image space. They are connected by a generative process $g : \mathcal{W} \rightarrow \mathcal{X}$, written as follows

$$x = g(w), x \in \mathcal{X}, w \in \mathcal{W}$$

(1)

In our setting, images are from different domains, where each domain represents a specific type of appearance condition, such as daytime in summer or nighttime in winter. We assume that the world space $\mathcal{W}$ can be decomposed into domain-unrelated space $\mathcal{S}$ and domain-related space $\mathcal{A}$, namely $\mathcal{W} = \mathcal{S} \times \mathcal{A}$. $\mathcal{S}$ contains place information (such as the structure of a building or shape of traffic sign) and $\mathcal{A}$ contains appearance information (such as sunny daytime in spring). Now, the generative process can be rewritten as

$$g : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{X}$$

$$x = g(s, a)$$

(2)

Any image $x \in \mathcal{X}$ can be seen as the composition of a domain-unrelated state $s \in \mathcal{S}$ and a domain-related state $a \in \mathcal{A}$ under $g$. We assume that states $s$ and $a$ are sampled independently from latent spaces $\mathcal{S}$ and $\mathcal{A}$, respectively.

Under the multi-domain setting, latent space $\mathcal{A}$ can be divided into different subspaces, such that

$$\mathcal{A} = \bigcup_{i=1}^{n} \mathcal{A}_i$$

(3)

where $n$ is the number of subspaces of $\mathcal{A}$. With $a_i \in \mathcal{A}_i$ sampled from one of these subspaces, the generated image $x_i$ can be written as

$$x_i = g(s, a_i) \in \mathcal{X}_i, s \in \mathcal{S}, a_i \in \mathcal{A}_i$$

(4)

Given $a_i$ and $a_j$ sampled from different subspaces, we assume that the generated images $x_i$ and $x_j$ follow different distributions or saying from different domains. We will use the notation $\mathcal{X}_i$ to denote a domain (i.e. the set of images collected from spring) in the rest of the article. Apparently, we have $n$ domains.

Images taken from the same place at different times with different appearances share the same $s$ and vary $a$. On the contrary, images collected continuously with similar appearances have identical $a$ and different $s$. 

Figure 1. The generative process and the inference process.
Inference process

Our goal is to find two functions $h_S$ and $h_A$ to extract disentangled representation $\hat{s}_i$ and $\hat{a}_i$:

$$\hat{s}_i = h_S(x_i), \hat{a}_i = h_A(x_i)$$ (5)

They are estimated from the inference processes $h_S$ and $h_A$, and we will refer to them as place feature and appearance feature, respectively. Apparently, we desire the place feature to be domain-unrelated and the appearance feature to be domain-related. Feature pair $z = (\hat{s}, \hat{a})$ is constituted of a place feature $\hat{s}$ and an appearance feature $\hat{a}$, and all the $z$s constitute the feature space $Z$.

As $s$ and $a_i$ in Eq. (4) are independent variables, $\hat{s}_i$ and $\hat{a}_i$ are also desired to be disentangled. Formally, a disentangled representation calls for that a change in $\hat{s}_i$ only corresponds to a change in $s$, while a change in $\hat{a}_i$ only corresponds to a change in $a_i$. In our setting, disentanglement requires that $\hat{s}_i$ and $\hat{a}_i$ should be unrelated. Additionally, these two features should have semantic meaning. For example, in our problem, it is required that $\hat{s}_i$ represents the place contents, such as the buildings and traffic signs in sight, while $\hat{a}_i$ represents the appearance information, such as the weather and the season. At an ideal state, a change in $\hat{s}_i$ should represent a change in place (i.e. moving 5 m forward), while a change in $\hat{a}_i$ should represent a change in appearance (i.e. shifting from sunny to rainy).

Adversarial disentangled feature learning

In this section, we present our feature disentanglement network in detail, including network architecture and the loss functions, which are summarized in Figure 2. At last, we introduce how to train the network and extend it to multiple domains.

Network architecture

The motivation of the proposed network is to extract disentangled features from given images. To achieve it, we set to constrain features explicitly to fulfill the disentanglement requirement.

Firstly, our method uses an autoencoder as the feature extractor. Autoencoder is a widely used self-supervised machine learning method, where the encoder encodes the input as a feature and the decoder tries to reconstruct the original input from the feature. Our encoder is composed of two parts, namely the place encoder $E_S$ and appearance encoder $E_A$, which is used to approximate $h_S$ and $h_A$, respectively. The decoder $G$ is used to reconstruct the input image, which ensures that the network learns meaningful features. Additionally, the dimension of the place and appearance features are usually smaller than the input dimension, leading to a more compact representation.
Pure autoencoder is not enough for disentanglement, as there is no constraint for \( \hat{s} \) and \( \hat{a} \) to be disentangled. As discussed above, we also need to ensure that a change in \( \hat{s} \) only corresponds to a change in \( s \), and a change in \( \hat{a} \) only corresponds to a change in \( a \). We solve this problem in two steps. Firstly, we introduce an appearance compatibility discriminator into the autoencoder, which is designed to ensure \( \hat{s} \) and \( \hat{a} \) are unrelated. Secondly, a place domain disriminator is designed to ensure \( \hat{s} \) that only has place information, leading to the disentanglement of \( \hat{s} \) and \( \hat{a} \). For simplicity, we start from two domains \((X_1, X_2)\)

\[
\hat{s}_1 = h_S(x_1), \hat{a}_1 = h_A(x_1) \\
\hat{s}_2 = h_S(x_2), \hat{a}_2 = h_A(x_2)
\]

where \( x_1 \) and \( x_2 \) are sampled from domains \( X_1 \) and \( X_2 \), respectively.

Our autoencoder follows the widely used bottleneck architecture, where the input is downsampled into two smaller feature maps by the encoders, while the decoder tries to recover the input from those two feature maps. The discriminators take combinations of those feature maps as input and give one dimension as output (Figure 2). The detailed architectures of different experiments are listed in the appendix.

**Autoencoder**

To measure the reconstruction quality, different distances can be used. In the previous version, we used L2 loss for reconstruction, which can be expressed as follows:

\[
L_{\text{recon}} = \frac{1}{2} \mathbb{E}_{x_i} || G(h_S(x_1), h_A(x_1)) - x_1 ||_2^2 \\
+ \frac{1}{2} \mathbb{E}_{x_i} || G(h_S(x_2), h_A(x_2)) - x_2 ||_2^2
\]

where \( x_i \) is an image sampled from some data distribution \( p(x_i) \) of domain \( X_i \). For specific tasks, such as place recognition, L2 loss is not the best choice. In this article, for the place recognition task, we try to further improve the reconstruction quality by constraining the reconstructed image to have similar high-level features to the original image. The high-level features are more meaningful because they have more “perceptual” information. Formally, this can be written as follows

\[
L_{\text{recon}} = \frac{1}{2} \mathbb{E}_{x_i} || F(G(h_S(x_1), h_A(x_1))) - F(x_1) ||_2^2 \\
+ \frac{1}{2} \mathbb{E}_{x_i} || F(G(h_S(x_2), h_A(x_2))) - F(x_2) ||_2^2
\]

where \( F \) is a pretrained VGG feature extractor. With this perceptual loss, the reconstructed images are more semantically meaningful than the ones with L2 loss, which also leads to more meaningful features. This helps for boosting place recognition performance, as shown in the experiments.

**Appearance compatibility discriminator**

As pointed out in the last section, disentanglement requires that \( \hat{s} \) only corresponds to \( s \) while \( \hat{a} \) only corresponds to \( a \). We firstly turn it into a “weaker” version that \( \hat{s} \) and \( \hat{a} \) are unrelated. As stated before, we want to constrain the features directly. We construct two tuples, \((\hat{s}_1, \hat{a}_1)\) and \((\hat{s}_2, \hat{a}_1)\), and constrain them to follow the same distribution. To fulfill this constraint, we design a discriminator, the appearance compatibility discriminator, which is denoted by \( D_{\text{app}} \). The input of \( D_{\text{app}} \) is the constructed tuples \((\hat{s}_1, \hat{a}_1)\) and \((\hat{s}_2, \hat{a}_1)\). We use an adversarial network to minimize the distance between these two tuples. The discriminator \( D_{\text{app}} \) and the encoders \( E_S \) and \( E_A \) are optimized successively. The encoders try to learn features, which minimize the distance between \((\hat{s}_1, \hat{a}_1)\) and \((\hat{s}_2, \hat{a}_1)\). On the other hand, the goal of \( D_{\text{app}} \) is to tell whether the given tuples follow the same distribution. In addition, instead of using Jensen–Shannon divergence as a distance metric, we use the Pearson \( \chi^2 \) divergence to get faster convergence, which uses L2 loss without sigmoid function.

During training, \( D_{\text{app}} \) and encoders \((E_S, E_A)\) are updated successively. The loss function for \( D_{\text{app}} \) is

\[
L_{\text{app}, 1} = \frac{1}{2} \mathbb{E}_{s_1, a_1} [(D_{\text{app}}(s_1, a_1) - 1)^2] \\
+ \frac{1}{2} \mathbb{E}_{s_2, a_1} [(D_{\text{app}}(s_2, a_1) - 0)^2]
\]

where \( s_1, \hat{s}_1, s_2, \hat{s}_2, a_1 \) and \( \hat{a}_1 \) are given by \( E_S(x_1), E_S(x_2), \) and \( E_A(x_1), x_1, x_2 \) are images from domain \( X_1, X_2 \), respectively.

Encoders \( E_S \) and \( E_A \) are encouraged to confuse \( D_{\text{app}} \). The loss function for autoencoder can be expressed as follows

\[
L_{\text{Adv}} = \frac{1}{2} \mathbb{E}_{s_1} [(D_{\text{app}}(E_S(x_1), E_A(x_1)) - 0)^2] \\
+ \frac{1}{2} \mathbb{E}_{s_2, x_2} [(D_{\text{app}}(E_S(x_2), E_A(x_1)) - 1)^2]
\]

It is worth noticing that \( E_S(x_1), E_S(x_2), \) and \( E_A(x_1) \) in Eq. (10) are in fact \( \hat{s}_1, \hat{s}_2, \) and \( \hat{a}_1 \) in Eq. (9). We replace them to remind that \( E_S \) is fixed when updating \( D_{\text{app}} \) (Eq. (9)), while \( D_{\text{app}} \) is fixed when updating \( E_S \) and \( E_A \) (Eq. (10)).

Equations (9) and (10) are formulated for the case that the first input image is sampled from \( X_1 \). When the first image is from \( X_2 \), we can derive \( L_{\text{Adv}, 2} \) and \( L_{\text{Adv}, 2} \) by exchanging two domains in Eqs. (9) and (10).
**Place domain discriminator**

With the appearance compatibility discriminator, we can only ensure that \( x \) and \( a \) are unrelated. An extreme situation is that \( x \) contains both place and appearance information, while \( a \) learns little information. This happens when the dimension of \( x \) is large enough to encode both \( x \) and \( a \). It often holds true when we do not know the actual dimension of \( x \) and \( a \), thus we have to set the dimension for \( x \) larger than \( a \). The reason is that place information is often much richer than appearance information for any given image. To address this problem, we construct another discriminator to ensure that \( x \) only contains place information. If satisfied, combined with \( D_{app} \), a change in \( x \) will only correspond to a change in \( a \). At the same time, all the appearance information will be represented in \( a \) with the reconstruction constraint. Thus, all the place features from \( X_1 \) and all the place features from \( X_2 \) are desired to follow the same distribution. When this holds true, it means neither \( x_1 \) nor \( x_2 \) is affected by domain-related states.

Supervised methods like NetVLAD constrain features from the same place with different appearances to be the same, which requires for alignment information. But as we hope to override the dependency on aligned data, we accomplish it in a different way. Constraining \( x_1 \) and \( x_2 \) to follow the same distribution is applicable using adversarial learning, but it is hard to extend to multiple domains. The reason is that such discriminator is domain-specific, and more domains require more discriminators. To overcome this, we construct the place feature pairs \((x_1, x_1')\) and \((x_2, x_2')\), where \( x_1' \) is another place feature from domain \( X_1 \). In particular, we sample another image \( x_1' \) from the domain \( X_1 \) and extract its place feature \( x_1 \). Then \((x_1, x_1')\) and \((x_2, x_2')\) are constrained to follow the same distribution, which implies that \( x_1 \) and \( x_2 \) follow the same distribution. Now we introduce the place domain discriminator, denoted by \( D_{pla} \). \( D_{pla} \) is designed to constrain \((x_1, x_1')\) and \((x_2, x_2')\) to follow the same distribution. Similar to the \( D_{app} \), the discriminator \( D_{pla} \) and the place encoder \( E_S \) are optimized in an adversarial way. Now we can formulate the loss function for \( D_{pla} \), which can be written as follows

\[
L_{D_{pla}}^{x_1,x_1'} = \frac{1}{2} \mathbb{E}_{x_1,x_1'}[(D_{pla}(x_1,x_1') - 1)^2] 
+ \frac{1}{2} \mathbb{E}_{x_2,x_2'}[(D_{pla}(x_1,x_1') - 0)^2] 
\]

where \( x_1 \) and \( x_1' \) are derived from \( E_S(x_1) \) and \( E_S(x_1') \), with \( x_1 \) and \( x_1' \) being two samples from the same domain \( X_1 \). The discriminator \( D_{pla} \) can be interpreted from another perspective: it outputs 1 if the given two-place features are from the same domain and 0 for those from different domains. Again, the loss function for the encoder \( E_S \) can be expressed as follows

\[
L_{Adv}^{x_1,x_1'} = \frac{1}{2} \mathbb{E}_{x_1,x_1'}[(D_{pla}(E_S(x_1),E_S(x_1')) - 0)^2] 
+ \frac{1}{2} \mathbb{E}_{x_1,x_1'}[(D_{pla}(E_S(x_1),E_S(x_1')) - 1)^2] 
\]

Similarly, we can have \( L_{D_{pla}}^{x_2,x_2'} \) and \( L_{Adv}^{x_2,x_2'} \).

**Training strategies**

The discriminators \( D_{pla} \) and \( D_{app} \) and encoders \( E_S \) and \( E_A \) are updated in turn by back-propagation. The only difference is that encoders are updated with the decoder \( G \) together. Each iteration of the training is constituted of three steps: (i) updating \( D_{pla} \), (ii) updating \( D_{app} \), and (iii) updating \( E_S, E_A, G \). Each training iteration can be formally written as follows

\[
\min_{D_{pla}} L_{D_{pla}}^{x_1,x_1'} \quad + \quad L_{D_{pla}}^{x_2,x_2'} 
\]

\[
\min_{D_{app}} L_{D_{app}}^{x_1,x_1'} \quad + \quad L_{D_{app}}^{x_2,x_2'} 
\]

\[
\min_{E_S,E_A,G} \lambda_1 (L_{Adv}^{x_1,x_1'} + L_{Adv}^{x_2,x_2'}) \quad + \quad \lambda_2 (L_{Adv}^{x_1,x_1'} + L_{Adv}^{x_2,x_2'}) 
\]

where \( \lambda_1 \) and \( \lambda_2 \) are hyperparameters to balance reconstruction and disentanglement, which is determined by grid search for different tasks. When training with only two domains, images \( x_1 \) and \( x_1' \) are sampled from the first domain, while \( x_2 \) and \( x_2' \) are from the second one. The network is trained with Adam optimizer, \( \beta_1 = 0.9 \) and \( \beta_2 = 0.999 \). Learning rates for autoencoder (including \( E_S, E_A, \) and \( G \)), \( D_{pla} \), and \( D_{app} \) are set as 0.000003, 0.00001, and 0.00001.

During training, the training datasets are augmented to increase robustness against viewpoint changes. For each image, we select four points on it randomly, and the surrounded area is cropped and wrapped into the original size. Besides, the wrapped image is flipped horizontally randomly.

**Extension: Multiple domain case**

Based on this domain-unrelated architecture, we present how to extend our network to multiple domains. Assume that there are \( n \) domains, denoted by \( X_1, X_2, \cdots, X_n \). In each training iteration, we firstly select two domains \( X_i \) and \( X_j \), randomly, where \( i, j = 1, 2, \cdots, n \) and \( i \neq j \). Then two batches of images are randomly sampled from \( X_i \) and \( X_j \), respectively. These images are fed into the network as training data for this iteration. During testing, when matching between any two of those domains, images are fed into \( E_S \) to obtain place features, followed by feature matching.
This extension does not require additional parameters. To see it, one should note that the autoencoder is shared across different domains. Besides, the discriminators are also shared and domain-unrelated, which only use information between two domains. For example, if the inputs of $D_{pla}$ are $(\hat{s}_i, \hat{s}_j)$ and $(\hat{s}_i, \hat{s}_j)$, $D_{pla}$ can be regarded as a metric to measure the distance between $(\hat{s}_i, \hat{s}_j)$ and $(\hat{s}_i, \hat{s}_j)$, instead of $\hat{s}_i$ and $\hat{s}_j$. It is similar to the triplet loss, which measures the metric distance between inner classes and outer classes. The parameters are shareable among different domains, and no additional parameters are needed when extending to more domains.

The extension enables that only one model is needed in a specific scene for long-term deployment. In the beginning, we have a baseline model trained from several domains. When new data with different appearances in the same area are available, the model can be fine-tuned by retraining on the dataset enhanced with newly collected data. The retraining does not require additional parameters. In contrast, style-transfer-based methods need new models to transfer new data into known styles. When new data come periodically, this will lead to quadratically increasing parameters. To see it, one can assume that there are $m$ domains. To transfer each domain to others, they need to train $\binom{m}{2} = \frac{1}{2}m(m-1)$ models. Thus compared to style-transfer-based methods, our method is more suitable for plugging into any long-term localization framework as the feature extractor.

**Experiments**

We conduct several experiments to illustrate our method. Firstly, we validate the proposed network with two toy cases, including Linear Gaussian for quantitative analysis and Colored MNIST for qualitative analysis. Then, we apply our network to visual place recognition and demonstrate its performance in different perspectives, including basic performance, ablation study, and multiple-domain performance.

**Toy case validation**

**Linear Gaussian.** We test the network on a linear gaussian generative process with two domains to validate whether it can produce disentangled representation as desired. The reason to choose the linear model is that we can use correlation as a quantitative metric for disentanglement in the linear case. The generative process can be written as follows

$$x = M_1s + M_2a_i, i = 1, 2$$

$$s \sim \mathcal{N}(\mu_s, \sigma_s^2) \in \mathbb{R}^{n_s}, a_i \sim \mathcal{N}(\mu_{a_i}, \sigma_{a,i}^2) \in \mathbb{R}^{n_a}$$

$$M_1 \in \mathbb{R}^{n_s \times n_a}, M_2 \in \mathbb{R}^{n_s \times n_a}$$

where $s$ and $a_i$ are vectors with the dimension of $n_s$ and $n_a$, and they are sampled from two normal distributions, respectively, with $\mu_s$ and $\mu_{a,i}$ as means and $\sigma_s^2$ and $\sigma_{a,i}^2$ as covariance matrices. Elements in $s$ are independent of each other, thus $\sigma_s^2$ is the diagonal matrix, so as $\sigma_{a,i}^2$. The generated data point $x$ is $n_s$-dimension vector. In our setting, $n_s > (n_s + n_a)$, thus each dimension in the generated data is not totally independent of others. This often happens in the natural world, such as pixels in an image, which are not independent of each other. Specifically, $n_s = 5$, $n_a = 5$, and $n_s = 50$.

The encoders $E_S$ and $E_{a,i}$ are single-layer linear modules. For another, the discriminators are multiple-layer nonlinear modules, with leaky rectified linear unit (LReLU) as an activation function.

To demonstrate the power of the two proposed adversarial losses, we compute the correlation matrix between the place feature $\hat{s}$ and appearance feature $\hat{a}_i$ of our method (Figure 3(c)). In comparison, we train a model with only an autoencoder (Figure 3(b)). Both of them are trained from the same initial state (Figure 3(a)). By comparing Figure 3(b) and (c), we can see that the introduced loss functions can produce less correlated $\hat{s}$ and $\hat{a}_i$. Besides, the
The correlation between place feature and appearance feature is nearly zero (Figure 3(c)), which shows that the features are disentangled very well. To further ensure that $D_{pla}$ is necessary, we train the network with only $D_{app}$ (Figure 3(d)).

From Figure 3(c) and (d), we can see that without $D_{pla}$, correlation is slightly larger. Thus, $D_{pla}$ can improve the disentanglement to some extent.

Figure 3(a) to (d) shows the case that $n_a = n_s = 5$ and $n_a = n_s = 5$, where $n_a$ and $n_s$ are the dimensions of appearance feature and place feature. As discussed above, in practice it is better to set larger $n_a$ and $n_s$ because we need to ensure $n_a > n_s$ and $n_s > n_a$, but $n_a$ and $n_s$ are unknown. To simulate the practical case, in Figure 3(e) to (h), we set $n_a = 10$ and $n_s = 10$. We get similar results to the former case, except that the autoencoder becomes more correlated when we increase the dimension of the feature space (Figure 3(b) and Figure (f)). When the features are over-parameterizing, purge autoencoder cannot guarantee that place feature and appearance feature are disentangled, and our method can effectively solve this problem.

**Colored MNIST.** The linear Gaussian is a simple toy case. To show the power of our method in a more complicated scene, we propose colored MNIST, an enhanced version of the handwritten digits dataset MNIST. In this dataset, the original grayscale digits are colored randomly in seven colors to simulate different domains. The colored MNIST is constructed to fulfill our hypothesis that the input data are composed of domain-related and domain-unrelated components. Based on this, we train our network and analyze the learned features qualitatively. Dimensions of features $n_a = 8$ and $n_s = 4 \times 4 \times 64 = 1024$. In training, $\lambda_1$ and $\lambda_2$ are set as 0.1.

To investigate what has the network learned from this toy case, we visualize the place feature and appearance feature using t-distributed stochastic neighbor embedding (t-SNE). As the dimensions of our features (1024 for $\hat{s}$ and 8 for $\hat{a}$) are too high for visualization, we employ t-SNE, which reduces high-dimensional data into two-dimensional ones for visualization. In this experiment, the “place” content is a digit, while the ‘appearance’ content is color. Each point in Figure 4(a) and (b) represents a sample from the colored MNIST dataset. For example, the spatial locations of all the points in Figure 4(a) represent the reduced place features of the samples. The color of each point in the left part of Figure 4(a) is determined the color of the sample, whereas the color of each point in the right part of Figure 4(a) is determined by the digit of the sample. Figure 4(b) is similar except that locations correspond to appearance features. From Figure 4(a), we can see that the place features can be easily clustered by digits, and they are unrelated to their color. It means that the place feature corresponds to the digit, and it is not affected by the color. Conversely, from Figure 4(b), we can find that the color information is already embedded into appearance features, and the same digit with different color can have varied appearance features. Thus, the appearance feature corresponds to the color, and it is not affected by the digit. From these
results, we can safely conclude that our network can obtain disentangled representation.

We also try to analyze our network in image space. As done in some image-to-image translation literature,67 we can replace the appearance features with those from another domain for the decoder to obtain image with different style. We implement this in two ways. Firstly, given two colored digits as inputs, we combine the place feature from the first digit and the appearance feature from the second digit and reconstruct a new digit through the decoder. The newly recovered digit is called translated image. Secondly, the appearance feature is replaced with a new vector filled with zeros, and the decoded image is called zero-appearance image. Results are presented in Figure 4(c). We sample images from every two domains. As there are seven domains in the dataset, there are \(7 \times 7\) subplots in Figure 4(c), except for the samples from the same domain (diagonal). Each subplot can be denoted by \(p_{ij}\), where \(i = 1, 2, \cdots, 7, j = 1, 2, \cdots, 7, i \neq j\). \(p_{ij}\) can be further divided by four parts, namely \(p_{ij}(m, n)\), where \(m = 1, 2, n = 1, 2\). For example, the two input images for \(p_{1,1}\) are a blue five \((p_{1,1}(1, 1))\) and a white seven \((p_{1,1}(1, 2))\). The translated image is a white five \((p_{1,1}(2, 1))\), while the zero-appearance image is a green five \((p_{1,1}(2, 2))\). We can see that the translated image keeps the shape of the first input image, while its color is determined by the second input image. It is also found that the colors of all the zero-appearance images are the same, while their digits remain the same as their original first input images. Thus, we can say that the digit of the reconstructed image is controlled by the place feature, while the color is determined by the appearance feature.

**Datasets**

To validate the proposed method on visual place recognition task, we test our network on three public datasets: Partitioned Nordland dataset,32 Alderley Day/Night dataset,7 and RobotCar-Seasons dataset.68 Besides, we also experiment on a new dataset, YQ Day/Night dataset, which is collected in a campus environment with changing appearance and will be opened to the public (https://tanigli.site/projects/academic/yq21).
**Partitioned Nordland dataset.** It is collected from a train on the same route (729 km) in four seasons (spring, summer, fall, and winter) with GPS data. In this article, four seasons are treated as four domains. The original Nordland dataset is used by many researches, but they use different partitions for training and testing. This dataset proposes a reasonable partition of Nordland, where the whole route is partitioned into five segments, with two as training set and three as testing set. The training and testing sets have 24,570 and 3450 images for each domain, respectively. The GPS information is accurate enough to align between different domains. This dataset provides images without perspective changes and high-quality ground truth, thus it is useful for validating the ability to overcome appearance changes (Figure 5(a)).

**Alderley Day/Night dataset.** It is captured from a camera mounted on a car. It is constituted of two domains, one daytime and one nighttime, on an 8 km journey. As mentioned in their article, the GPS data are not reliable enough, thus the images are manually aligned frame by frame. However, as the images are collected in an urban environment, the vehicle is moving with lateral and heading changes, which makes the alignment less reliable. Besides, the apparent differences between those two sessions are really large, making it challenging for place recognition. From Figure 5(b), one can see that sometimes it is even challenging for humans. Each domain in the dataset is split as a training and testing set with 10,007 and 4600 images, respectively.

**RobotCar-Seasons dataset.** It is based on the Oxford RobotCar dataset, which is recorded on a vehicle with six cameras under different conditions in the urban environment, with other sensors including INS, GPS, and LiDAR. This dataset is collected from a camera mounted on a car. It is constituted of two domains, one daytime and one nighttime, on an 8 km journey. As mentioned in their article, the GPS data are not reliable enough, thus the images are manually aligned frame by frame. However, as the images are collected in an urban environment, the vehicle is moving with lateral and heading changes, which makes the alignment less reliable. Besides, the apparent differences between those two sessions are really large, making it challenging for place recognition. From Figure 5(b), one can see that sometimes it is even challenging for humans. Each domain in the dataset is split as a training and testing set with 10,007 and 4600 images, respectively.

**Evaluation metrics and training details**

We choose two widely used metrics for visual place recognition in the following experiments (except for RobotCar-Seasons which will be discussed later): area under curve (AUC) and accuracy (true positive rate). After the training is done, the place features are used as global descriptors for matching. The goal of visual place recognition is to find the nearest image to a given query image $x_Q$ in database $NDB$. The query set is further split as all-day and all-night, where all-day refers to images collected in the daytime, while all-night refers to images collected in the nighttime. In this article, only the all-night subset is used, as we want to validate our method in different domains. The ground truth is obtained from large-scale structure from motion and initialized with INS data. One thing that should be mentioned is that the ground truth is only used in the testing phase, not in the training process. This dataset provides challenges in autonomous driving including appearance changes, perspective difference, and motion blur (Figure 5(c)).

**YQ Day/Night dataset.** It is a subset of the YQ dataset. The original YQ dataset has 21 sessions, out of which we choose two for the place recognition task in this article. These two sessions become two domains in this dataset, namely day and night. The first session is collected in the morning, while the second one is collected in the evening (Table 1). The evening traversal is collected at the time when it is turning from day to night. Although this traversal only lasts for about 19 min, it contains a different appearance. Thus, it can be used to validate the robustness of algorithms against dynamic appearance changes in a short period. Sample images are shown in Figure 5(d). The ground truth is obtained from the LiDAR SLAM results. We split the trajectory into two folds, with the 60% beginning being the training set and the 30% ending as the testing set. The training set is sampled every 0.1 m, and the testing set is sampled every 2 m. As the data are recorded on a mobile robot controlled by a remoter at low speed, it can be seen as a representative in mobile logistics in a small area, with appearance and perspective changes.

| Session | Date and time | Duration | Number of images |
|---------|---------------|----------|------------------|
| Day     | 2017/03/03 07:52:31 | 17.7 min | 5393 181 |
| Night   | 2017/03/07 18:07:21 | 19.8 min | 4973 181 |

Table 1. Details of YQ Day/Night dataset.
100 epochs, except for that model of the YQ Day/Night dataset, which is trained for 50 epochs. It is because the number of images in YQ is too small, and the model gets overfitted too quickly.

Comparison methods. To illustrate the performance of our method on visual place recognition, several methods are selected as a comparison:

- Three methods based on handcrafted features (DBoW2, HOG, and DenseVLAD) are chosen as representatives of traditional methods.24,27,55
- For supervised methods, NetVLAD and method by Facil et al. are used to show their advantages and limitation.10,32 As NetVLAD has opened the code for training NetVLAD, we also retrain NetVLAD on these datasets to see the improvement obtained from supervision. All settings follow the original article of NetVLAD, except that when training YQ Day/Night, the “nNegChoice,” “nTestSample,” and “nTestRankSample” are set as 20, 20, and 100, respectively, because the dataset is small.
- Besides, two style-transfer-based algorithms are also selected.12,71

RobotCar-Seasons dataset uses different evaluation metrics. As described by the benchmark, RobotCar-Seasons dataset measures the performance using percentages of query images localized within three error tolerance thresholds.68

### Ablation study

Our method introduces two new adversarial losses to the autoencoder. To see whether the new losses work, we conduct an ablation study in this section. We selected the most challenging pair in Partitioned Nordland, namely winter and spring, as target domains. The experiment follows the same settings described in the last subsection, except the hyperparameters $\lambda_1$ and $\lambda_2$. The baseline is the autoencoder, where $\lambda_1$ and $\lambda_2$ are set as zeros. To see the necessity of $D_{pla}$ and $D_{app}$, $D_{app}$ is added to the baseline ($\lambda_1 = 0.003$ and $\lambda_2 = 0$), followed by adding $D_{pla}$ ($\lambda_1 = 0.003$ and $\lambda_2 = 0.01$). Finally, augmentation is added as a comparison.

Results are displayed in Table 2. We can see that the complete network (with $D_{pla}$, $D_{app}$, and augmentation) is much better than the baseline (pure autoencoder). By adding $D_{app}$, the network gets slight improvement (0.06 in AUC and 9.9 percent points in accuracy), while by adding $D_{pla}$ further, more improvements are obtained (0.21 in AUC and 11.5 percent points in accuracy). Finally, by adding augmentation, we can have a further performance improvement. Thus, we can safely conclude that the introduced losses are effective in visual place recognition.

### Two domains

This section investigates the performances of different methods in the scenario with two domains. For Partitioned Nordland dataset, only the winter and spring sessions are chosen as two domains.

Experimental results are listed in Tables 3 and 4. Table 3 presents the performance of different methods on three of the mentioned datasets: Partitioned Nordland, Alderley Day/Night, and YQ Day/Night. To evaluate our method on RobotCar-Seasons dataset, we submit our results to the Visual Localization Benchmark (https://www.visuallocalization.net/benchmark). In Table 4, we also consider several methods on that benchmark as a comparison (only published place recognition algorithms are selected).

#### Table 2. Ablation study.

| $D_{pla}$ | $D_{app}$ | Augmentation | AUC | Accuracy |
|-----------|-----------|--------------|-----|----------|
| $\times$  | $\times$  | $\times$     | 0.50| 34.1%    |
| $\checkmark$ | $\times$ | $\times$ | 0.53| 36.1%    |
| $\checkmark$ | $\checkmark$ | $\times$ | 0.66| 48.8%    |
| $\checkmark$ | $\checkmark$ | $\checkmark$ | 0.87| 66.3%    |

AUC: area under curve.

#### Table 3. Performance of different methods.

| Method       | Need aligned data? | Partitioned Nordland |   | Alderley Day/Night |   | YQ Day/Night |   |
|--------------|--------------------|-----------------------|---|-------------------|---|-------------|---|
|              |                    | AUC  | Accuracy     | AUC | Accuracy | AUC | Accuracy |
| DBoW2        | $\times$           | 0.09 | 1.33%        | 0.00 | 0.22%   | 0.071| 4.97%    |
| HOG          | $\times$           | 0.17 | 17.0%        | 0.02 | 1.85%   | 0.044| 4.42%    |
| Merrill et al.11 | $\times$ | 0.28 | 15.8%        | 0.10 | 1.26%   | 0.174| 14.92%   |
| NetVLAD      | $\times$           | 0.22 | 19.9%        | 0.02 | 2.65%   | 0.224| 30.39%   |
| ToDayGAN     | $\times$           | 0.10 | 13.4%        | 0.02 | 1.79%   | 0.162| 22.10%   |
| Ours         | $\times$           | 0.87 | 66.3%        | 0.45 | 25.5%   | 0.610| 42.5%    |
| Ours-L2      | $\times$           | 0.70 | 49.4%        | 0.34 | 21.0%   | 0.510| 40.9%    |
| Facil et al.32 | $\checkmark$ | N/A1 | 92%          | N/A1| 7.82%   | N/A1| N/A1     |
| NetVLAD      | $\checkmark$      | 0.74 | 83.0%        | 0.13 | 15.8%   | 0.078| 11.05%   |

AUC: area under curve; HOG: histograms of oriented gradient; NA: not available because they do not use AUC as criteria.
From Table 3, we can see that methods based on hand-crafted features are not good at place recognition tasks with extreme appearance changes. For learning-based, the supervised methods outperform self-supervised ones in some dataset (Partitioned Nordland) as expected. However, our method is better than those supervised methods on some datasets (Alderley Day/Night and YQ Day/Night). There are two possible reasons. One is that supervised methods depend on the quality of alignment. As we described before, the alignment of Alderley Day/Night is not good enough somewhere, which may degrade the performance for supervised methods. Another reason is that the training set and testing set of YQ Day/Night have subtle differences as described above, which request for generalization ability for algorithms. Generally speaking, self-supervised methods outperform supervised methods in the sense of generalization.

Compared with other self-supervised methods, our method achieves comparable results. It shows the best performance on Partitioned Nordland, Alderley Day/Night, and YQ Day/Night. Combine with Table 4, we can find that our method and ToDayGAN perform better than other self-supervised methods on different datasets. One main difference between our method and ToDayGAN is that in our method the adversarial learning is applied directly to features, while ToDayGAN is on images. The transferred image by ToDayGAN looks realistic, but sometimes it still looks different from the original style image (the reader can see transferred images in the appendix). The reason is that in style transfer the same input image can generate different output. Especially, Partitioned Nordland has a slight appearance difference in the same domain, resulting in the miss-matching problem. Conversely, the appearance of the same domain in RobotCar-Seasons is quite stable, that is why ToDayGAN outperforms our method on RobotCar-Seasons in some criteria.

To see the improvement brought by perceptual loss, we also trained our network with L2 reconstruction loss (Our-L2 in Table 3). From Table 3, we can see that perceptual loss has an obvious increase compared to L2 loss. This

| Method         | 0.25 m | 0.5 m | 5 m | 10 m |
|----------------|--------|-------|-----|------|
| NetVLAD        | 0.3    | 2.3   | 15.9|
| ToDayGAN       | 2.2    | 10.8  | 50.5|
| DenseVLAD      | 1.0    | 4.4   | 22.7|
| Hu et al.71    | 2.5    | 6.5   | 15.8|
| Ours           | 3.3    | 13.4  | 46.5|

Red: best performance; blue: second-best performance.
finding illustrates that the learned features can improved by reconstructing high-level features instead of reconstructing original images.

We also try to explore what does the network learns by generating images from cross-domain features. For specific, we feed place features from one domain (Figure 6(a)) with appearance features from the other domain (Figure 6(b)) into our network to see what will be generated from the decoder $G$ (Figure 6(c)). In Figure 6(c), one can find that the place information (e.g. tracks, buildings, and traffic lights) is determined by images in the first row (Figure 6(a)), while the appearance information (e.g. color of the ground, illumination) is determined by the second row (Figure 6(b)). This result satisfies our motivation: the place information is embedded in the place feature, while the appearance feature controls the appearance of the reconstructed images. To further illustrate what is learned in the place feature, we replace all the appearance features with all-zero vectors (Figure 6(d)). All the zero-appearance images look to have a similar appearance, while their place information remains the same as the first row (Figure 6(a)). These two findings demonstrate that the proposed method can disentangle the input image across appearance changes.

**Multiple domains**

One novelty of our network is that it is designed to be trainable with multiple domains without additional parameters. The benefit is that when new data from different domain come, we can retrain our network without increasing model capacity. We use Partitioned Nordland dataset to illustrate this point as it has four domains. Firstly, for every two domains, we train a network and evaluate it on the testing set. In this process, we have 12 models in total. Secondly, we train a unified model with four domains as the training set and then evaluate it on every two domains. In this stage, only one model is obtained.

Table 5 is the comparison results between two-domain models and multiple-domain models. We can see that by fusing more domains, our network achieves comparable performance to the two-domain models. However, the two-domain network needs 12 models, while the multiple-domain network needs only 1 model. It means our method can be extended to more domains while keeping the same model complexity. This is very useful when deploying deep learning networks in an environment with changing appearance.

**Conclusion**

We propose a feature disentanglement network for place recognition, which is composed of an autoencoder and two discriminators. By training the network in an adversarial manner, we can obtain domain-unrelated and domain-related features from multiple-domain data. The appearance compatibility discriminator enforces the appearance feature to be invariant to place content, while the place domain discriminator constrains the place feature to be robust against appearance changes. Qualitative and quantitative results on two two-domain cases demonstrate that our network is capable to obtain disentangled representation. Experiments on four datasets show that our method achieves favorable performance in visual place recognition tasks. Additionally, our network can be extended to multiple domains without increasing model capacity and sacrificing performance. We also open a new place recognition dataset for the research community.
Appendices

Network architecture

Networks for the linear case, colored MNIST, and place recognition are listed in Tables 6 to 8, respectively. Conv-\((N_n, K_k, P_p, S_s)\) denotes a convolution with output channels of \(n\), kernel size of \(k\), padding of \(p\), and stride of \(s\). Two types of pooling are used, including max-pooling (MaxPool) of kernel size 2 \(\times\) 2 and average-pooling (AvgPool) of kernel size 8 \(\times\) 8. Linear-\((n)\) represents a fully connected layer with \(n\) outputs. Decoders upsample feature maps using the nearest neighbor algorithm (upsample).

Each layer can be optionally followed by normalization function and activation function. Normalization functions include instance normalization\(^7\) and layer normalization\(^3\). Activation functions include rectified linear unit (ReLU), LReLU (with a negative slope of 0.2), and Tanh. The concat operation tries to fuse two feature maps. For input with the same spatial dimensions, such as \(H \times W \times C_1\) and \(H \times W \times C_2\), the output of concat is determined by concatenating these two features along the last axis. Thus, the size of the output is \(H \times W \times (C_1 + C_2)\). If input feature maps have different spatial dimensions, such as \(H \times W \times C_1\) and \(1 \times 1 \times C_2\), the smaller one will be repeated along two spatial axes firstly to match with the larger one, resulting a \(H \times W \times C_2\) feature map. Then, it is concatenated with the larger one, with a \(H \times W \times (C_1 + C_2)\) feature map as output.

Samples of style transfer

Figure 7 is the sample output by Anoosheh et al.\(^1\), including three datasets (Partitioned Nordland, Alderley Day/Night, and YQ Day/Night).

| Layers | Appearance compatibility discriminator | Output size |
|-------|---------------------------------------|-------------|
| 0     | Concat-(8 \(\times\) 8 \(\times\) 64, 1 \(\times\) 1 \(\times\) 8) | 8 \(\times\) 8 \(\times\) 72 |
| 1     | Conv-(N128, K4, P1, S2), LN, LReLU      | 4 \(\times\) 4 \(\times\) 128 |
| 2     | Conv-(N256, K4, P1, S2), LN, LReLU      | 2 \(\times\) 2 \(\times\) 256 |
| 3     | Conv-(N512, K4, P1, S2), LN, LReLU      | 1 \(\times\) 1 \(\times\) 512 |
| 4     | Conv-(N64, K1, P0, S1), LyReLU          | 1 \(\times\) 1 \(\times\) 64 |
| 5     | Conv-(N1, K1, P0, S1)                  | 1 \(\times\) 1 \(\times\) 1 |

ReLU: rectified linear unit; IN: instance normalization; LN: layer normalization.
Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed the receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported in part by the National Nature Science Foundation of China under Grant 61903332, and in part by the Natural Science Foundation of Zhejiang Province under grant number LGG21F030012.

ORCID iD

Li Tang  https://orcid.org/0000-0003-2590-6872

References

1. Bresson G, Alsayed Z, Yu L, et al. Simultaneous localization and mapping: a survey of current trends in autonomous driving. IEEE Trans Intell Veh 2017; 2(3): 194–220.
2. Cadena C, Carlone L, Carrillo H, et al. Past, present, and future of simultaneous localization and mapping: toward the robust-perception age. IEEE Trans Robot 2016; 32(6): 1309–1332.
3. Lowe DG. Distinctive image features from scale-invariant keypoints. Int J Comput Vis 2004; 60(2): 91–110.
4. Rublee E, Rabaud V, Konolige K, et al. ORB: An efficient alternative to SIFT or SURF. In International conference on computer vision, 2011, pp. 2564–2571. DOI: 10.1109/ICCV.2011.6126544.
5. Schindler G, Brown M, and Szeliski R. City-scale location recognition. In 2007 IEEE conference on computer vision and pattern recognition, Minneapolis, MN, 17 June 2007 to 22 June 2007, pp. 1–7. IEEE.
6. Cummins M and Newman P. FAB-MAP: probabilistic localization and mapping in the space of appearance. Int J Robot Res 2008; 27(6): 647–665.
7. Milford MJ and Wyeth GF. SeqSLAM: visual route-based navigation for sunny summer days and stormy winter nights. In 2012 IEEE international conference on robotics and automation, St Paul, MN, USA, 14 May 2012–19 May 2012, pp. 1643–1649. IEEE.
8. Doan AD, Latif Y, Chin TJ, et al. Scalable place recognition under appearance change for autonomous driving. In Proceedings of the IEEE international conference on computer vision, Long Beach, CA, 16–20 June 2019, pp. 9319–9328.
9. Chen Z, Jacobson A, Sudderhauf N, et al. Deep learning features at scale for visual place recognition. In 2017 IEEE international conference on robotics and automation (ICRA), Marina Bay Sands, Singapore, From 29 May 2017 until 3 June 2017, pp. 3223–3230. IEEE.
10. Arandjelovic R, Gronat P, Torii A, et al. NetVLAD: CNN architecture for weakly supervised place recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, Las Vegas, 27–30 June 2016, pp. 5297–5307.
11. Merrill N and Huang G. Lightweight unsupervised deep loop closure. In Proceedings of robotics: science and systems (RSS), Pittsburgh, PA, 2018. (accepted).
12. Anosheh A, Sattler T, Timofte R, et al. Night-to-day image translation for retrieval-based localization. In 2019 International conference on robotics and automation (ICRA), Montreal, Canada, 20–24 May 2019, pp. 5958–5964. IEEE.
13. Lee HY, Tseng HY, Mao Q, et al. DRIT++: diverse image-to-image translation via disentangled representations. Int J Comp Vis 2020; 128: 2402–2417.
14. Higgins I, Amos D, Pfau D, et al. Towards a definition of disentangled representations. arXiv preprint arXiv: 1812.02230. 2018.
15. Tang L, Wang Y, Luo Q, et al. Adversarial feature disentanglement for place recognition across changing appearance. In International conference on robotics and automation (ICRA), p. accepted, Paris, France, 31 May–15 June 2020, pp. 1301–1307. IEEE.
16. Li F and Kosecka J. Probabilistic location recognition using reduced feature set. In Proceedings 2006 IEEE international conference on robotics and automation, 2006. ICRA 2006, Orlando, FL, USA, 5–19 May 2006, pp. 3405–3410. IEEE.
17. Nowicki MR, Wietrzykowski J, and Skrzypczynski P. Real-time visual place recognition for personal localization on a mobile device. Wirel Pers Commun 2017; 97(1): 213–244.
18. Zaffar M, Ehsan S, Milford M, et al. Memorable maps: a framework for re-defining places in visual place recognition. IEEE Trans Intell Transp Syst 2020; 1–15. DOI:10.1109/TITS.2020.3001228.
19. Schlegel D and Grisetti G. HBST: a hamming distance embedding binary search tree for feature-based visual place recognition. IEEE Robot Auto Lett 2018; 3(4): 3741–3748.
20. Kanji T. Mining minimal map-segments for visual place classifiers. arXiv preprint arXiv:190900594. 2019.
21. Yin P, Srivatsan RA, Chen Y, et al. MRS-VPR: a multi-resolution sampling based global visual place recognition method. In 2019 International conference on robotics and automation (ICRA), Montreal, Canada, May 20–24 2019, pp. 7137–7142. IEEE.
22. Hausler S and Milford M. Hierarchical multi-process fusion for visual place recognition. In 2020 IEEE International conference on robotics and automation, ICRA 2020, Paris, France, May 31–August 31 2020, pp. 3327–3333. IEEE. DOI:10.1109/ICRA40945.2020.9197360. URL https://doi.org/10.1109/ICRA40945.2020.9197360.
23. Schubert S, Neubert P, and Protzel P. Unsupervised learning methods for visual place recognition in discrete and continuously changing environments. In 2020 IEEE international conference on robotics and automation (ICRA), Paris, France, 31 May 2020–31 August 2020, pp. 4372–4378. DOI:10.1109/ICRA40945.2020.9197044.
24. Dalal N and Triggs B. Histograms of oriented gradients for human detection. In 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR’05), San
25. Murillo AC, Singh G, Kosecká J, et al. Localization in urban environments using a panoramic GIST descriptor. *IEEE Trans Robot* 2013; 29(1): 146–160.

26. Mur-Artal R, Montiel JMM, and Tardos JD. ORB-SLAM: a versatile and accurate monocular slam system. *IEEE Trans Robot* 2015; 31(5): 1147–1163.

27. Gálvez-López D and Tardos JD. Bags of binary words for fast place recognition in image sequences. *IEEE Trans Robot* 2012; 28(5): 1188–1197.

28. Jégou H, Douze M, Schmid C, et al. Aggregating local descriptors into a compact image representation. In *CVPR 2010-23rd IEEE conference on computer vision & pattern recognition*, San Francisco, CA, USA, 13–18 June 2010, pp. 3304–3311. IEEE Computer Society.

29. Krizhevsky A, Sutskever I, and Hinton GE. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 2012. pp. 1097–1105.

30. Sünderhauf N, Shirazi S, Dayoub F, et al. On the performance of convnet features for place recognition. In *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, Hamburg, Germany, 28 September–3 October 2015, pp. 4297–4304. IEEE.

31. Deng J, Dong W, Socher R, et al. ImageNet: a large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, Miami, FL, June 20 2009 to June 25 2009, pp. 248–255. IEEE.

32. Facil JM, Olid D, Montesano L, et al. Condition-invariant multi-view place recognition. *arXiv preprint arXiv: 190209516*. 2019.

33. Hausler S, Jacobson A, and Milford M. Filter early, match late: improving network-based visual place recognition. In *2019 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, Macao, China, 03 November 2019–08 November 2019, pp. 3268–3275. DOI:10.1109/IROS40897.2019.8967783.

34. Pillai S and Leonard J. Self-supervised visual place recognition learning in mobile robots. In: *Learning for localization and mapping workshop at IROS 2017 IROS: IEEE/RSJ international conference on intelligent robots and systems (IROS)*, Vancouver, BC, Canada, 24–28 September 2017.

35. Yu J, Zhu C, Zhang J, et al. Spatial pyramid-enhanced NetVLAD with weighted triplet loss for place recognition. *IEEE Trans Neur Netw Learn Syst* 2019; 31: 661–674.

36. Sünderhauf N, Shirazi S, Jacobson A, et al. Place recognition with convnet landmarks: viewpoint-robust, condition-robust, training-free. In *Proceedings of robotics: science and systems XII*, 2015.

37. Hong Z, Petillot Y, Lane D, et al. Textplace: visual place recognition and topological localization through reading scene texts. In *Proceedings of the IEEE international conference on computer vision*, Seoul, Korea, 27 October–2 November, pp. 2861–2870.

38. Khaliq A, Ehsan S, Chen Z, et al. A holistic visual place recognition approach using lightweight cnns for significant viewpoint and appearance changes. *IEEE Trans Robot* 2019; 36(2): 1–9.

39. Chen Z, Liu L, Sa I, et al. Learning context flexible attention model for long-term visual place recognition. *IEEE Robot Auto Lett* 2018; 3(4): 4015–4022.

40. Xin Z, Cai Y, Lu T, et al. Localizing discriminative visual landmarks for place recognition. In *2019 International conference on robotics and automation (ICRA)*, Montreal, Canada, 20–24 May 2019, pp. 5979–5985. IEEE.

41. Khaliq A, Ehsan S, Milford M, et al. Camal: context-aware multi-scale attention framework for lightweight visual place recognition. *arXiv preprint arXiv:190908153*. 2019.

42. Lowry S and Milford MJ. Supervised and unsupervised linear learning techniques for visual place recognition in changing environments. *IEEE Trans Robot* 2016; 32(3): 600–613.

43. Garg S and Milford M. Fast, compact and highly scalable visual place recognition through sequence-based matching of overloaded representations. In *2020 IEEE international conference on robotics and automation (ICRA)*, Paris, France, 31 May 2020–31 Aug 2020, pp. 3341–3348. DOI:10.1109/ICRA40945.2020.9196827.

44. Hinton GE and Salakhutdinov RR. Reducing the dimensionality of data with neural networks. *Sci 2006; 313(5786): 504–507.

45. Shantia A, Timmers R, Schomaker L, et al. Indoor localization by denoising autoencoders and semi-supervised learning in 3d simulated environment. In *2015 international joint conference on neural networks (IJCNN)*, Killarney, Ireland, 11 July 2015–16 July 2015, pp. 1–7. IEEE.

46. Mukherjee A, Chakraborty S, and Saha SK. Learning deep representation for place recognition in slam. In *International Conference on pattern recognition and machine intelligence, Pattern Recognition and Machine Intelligence*, 2017, pp. 557–564. Springer.

47. Gao X and Zhang T. Unsupervised learning to detect loops using deep neural networks for visual slam system. *Auto Robot* 2017; 41(1): 1–18.

48. Liu MY, Breuel T, and Kautz J. Unsupervised image-to-image translation networks. In *Advances in neural information processing systems*, 2017. pp. 700–708.

49. Isola P, Zhu JY, Zhou T, et al. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, Honolulu, Hawaii, 21–26 July 2017, pp. 1125–1134.

50. Goodfellow I, Pouget-Abadie J, Mirza M, et al. Generative adversarial nets. In *Advances in neural information processing systems*, 2014. pp. 2672–2680.

51. Porav H, Maddern W, and Newman P. Adversarial training for adverse conditions: robust metric localisation using appearance transfer. In *2018 IEEE international conference on robotics and automation (ICRA)*, Brisbane, Australia, 21 May 2018–25 May 2018, pp. 1011–1018. IEEE.
52. Latif Y, Garg R, Milford M, et al. Addressing challenging place recognition tasks using generative adversarial networks. In 2018 IEEE international conference on robotics and automation (ICRA), Brisbane, Australia, 21 May 2018–25 May 2018, pp. 2349–2355. IEEE.

53. Clement L and Kelly J. How to train a cat: learning canonical appearance transformations for direct visual localization under illumination change. *IEEE Robot Auto Lett* 2018; 3(3): 2447–2454.

54. Yin P, Xu L, Li X, et al. A multi-domain feature learning method for visual place recognition. In 2019 *International conference on robotics and automation (ICRA)*, Montreal, Canada, 20 May 2019–24 May 2019, pp. 319–324. IEEE.

55. Torii A, Arandjelovic R, Sivic J, et al. 24/7 place recognition by view synthesis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, Boston, Massachusetts, 8–10 June 2015, pp. 1808–1817.

56. Garg S, Suenderhauf N, and Milford M. Semantic–geometric visual place recognition: a new perspective for reconciling opposing views. *Int J Robot Res.* Epub ahead of print 8 April 2019. DOI: 10.1177/0278364919839761.

57. Johnson J, Alahi A, and Fei-Fei L. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, Amsterdam, Netherlands, 8–16 October 2016.

58. Simonyan K and Zisserman A. Very deep convolutional networks for large-scale image recognition. In: Y Bengio and Y LeCun (eds) *3 rd International conference on learning representations, ICLR 2015*, San Diego, CA, USA, 7–9 May 2015, Conference Track Proceedings. URL http://arxiv.org/abs/1409.1556.

59. Mao X, Li Q, Xie H, et al. Least squares generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, Venice, Italy, 22–29 October 2017, pp. 2794–2802.

60. Kingma DP and Ba J. ADAM: a method for stochastic optimization. In: Y Bengio and Y LeCun (eds) *3 rd International conference on learning representations, ICLR 2015*, San Diego, CA, USA, 7–9 May 2015, Conference Track Proceedings. URL http://arxiv.org/abs/1412.6980.

61. Schroff F, Kalenichenko D, and Philbin J. FaceNet: a unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, Boston, 7–12 June 2015, pp. 815–823.

62. Tang L, Wang Y, Ding X, et al. Topological local-metric framework for mobile robots navigation: a long term perspective. *Auto Robot* 2019; 43(1): 197–211.

63. Churchill W and Newman P. Experience-based navigation for long-term localisation. *Int J Robot Res* 2013; 32(14): 1645–1661.

64. Maas AL, Hannun AY, and Ng AY. Rectifier nonlinearities improve neural network acoustic models. In *Proceedings of the ICML*, Atlanta, Georgia, USA, 16–23 June 2013.

65. LeCun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition. *Proc IEEE* 1998; 86(11): 2278–2324.

66. Maaten Lvd and Hinton G. Visualizing data using t-SNE. *J Mach Learn Res* 2008; 9(Nov): 2579–2605.

67. Huang X, Liu MY, Belongie S, et al. Multimodal unsupervised image-to-image translation. In *Proceedings of the European conference on computer vision (ECCV)*, Munich, Germany, 8–14 September 2018, pp. 172–189.

68. Sattler T, Maddern W, Toft C, et al. Benchmarking 6DOF outdoor visual localization in changing conditions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, Salt Lake City, Utah, 19–21 June 2018, pp. 8601–8610.

69. Gomez-Ojeda R, Lopez-Antequera M, Petkov N, et al. Training a convolutional neural network for appearance-invariant place recognition. *arXiv preprint arXiv:150507428*. 2015.

70. Maddern W, Pascoe G, Linegar C, et al. 1 Year, 1000 km: the oxford robotcar dataset. *Int J Robot Res* 2017; 36(1): 3–15.

71. Hu H, Wang H, Liu Z, et al. Retrieval-based localization based on domain-invariant feature learning under changing environments. In 2019 *IEEE/RSJ international conference on intelligent robots and systems (IROS)*, Macau, China, 3 November 2019 until 8 November 2019. pp. 3684–3689. DOI:10.1109/IROS40897.2019.8968047.

72. Ulyanov D, Vedaldi A, and Lempitsky V. Improved texture networks: maximizing quality and diversity in feed-forward stylization and texture synthesis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, Honolulu, Hawaii, 21–26 July 2017, pp. 6924–6932.

73. Lei Ba J, Kiros JR, and Hinton GE. Layer normalization. *arXiv preprint arXiv:160706450*. 2016.