Learning Transformation-Predictive Representations for Detection and Description of Local Features

Zihao Wang[1] Chunxu Wu[2] Yifei Yang[2] Zhen Li[2]*
[1]School of Intelligence Science and Technology, Peking University
[2]School of Automation, Beijing Institute of Technology
zhwang@stu.pku.edu.cn, {3120200960, 3120220892, zhenli}@bit.edu.cn

Abstract

The task of key-points detection and description is to estimate the stable location and discriminative representation of local features, which is a fundamental task in visual applications. However, either the rough hard positive or negative labels generated from one-to-one correspondences among images may bring indistinguishable samples, like false positives or negatives, which acts as inconsistent supervision. Such resultant false samples mixed with hard samples prevent neural networks from learning descriptions for more accurate matching. To tackle this challenge, we propose to learn the transformation-predictive representations with self-supervised contrastive learning. We maximize the similarity between corresponding views of the same 3D point (landmark) by using none of the negative sample pairs and avoiding collapsing solutions. Furthermore, we adopt self-supervised generation learning and curriculum learning to soften the hard positive labels into soft continuous targets. The aggressively updated soft labels contribute to overcoming the training bottleneck (derived from the label noise of false positives) and facilitating the model training under a stronger transformation paradigm. Our self-supervised training pipeline greatly decreases the computation load and memory usage, and outperforms the sota on the standard image matching benchmarks by noticeable margins, demonstrating excellent generalization capability on multiple downstream tasks.

1. Introduction

Local visual descriptors are fundamental to various computer vision applications such as camera calibration [37], 3D reconstruction[19], visual simultaneous localization and mapping (VSLAM) [33], and image retrieval [38]. The descriptors indicate the representation vector of the patch around the key-points and can be used to generate dense correspondences between images.

The discriptors are highly dependent on the effective representation, which has been always trained with the Siamese architecture and contrastive learning loss [12, 29, 39]. The core idea of contrastive learning is "learn to compare": given an anchor key-point, distinguish a similar (or *positive*) sample from a set of dissimilar (or *negative*) samples, in a projected embedding space. The induced representations present two key properties: 1) *alignment* of features from positive pairs, 2) and *uniformity* of representation on the hypersphere [51]. Negative samples are thus introduced to keep the uniformity property and avoid model collapse, *i.e.*, preventing the convergence to one constant solution [13]. Therefore, various methods have been proposed to mine hard negatives [6, 21, 55]. However, these methods raise the computational load and memory resources usage heavily [17]. More Importantly, within the hard negatives, some samples are labeled as negatives, but actually have the identical semantics of the anchor (*i.e.*, *false negatives*). These false negatives act as inconsistent supervision and prevent the learning-based models from achieving higher accuracy [4]. More concretely, the false negatives represent the instance located on the repetitive texture in the structural dataset, as shown in Figure 1. It is challenging to recognize such false negatives from true negatives [39].

The recent active self-supervised learning methods [5, 8, 9, 15] motivate us to rethink the effectiveness of negatives in descriptors learning. We propose to learn the transformation predictive representations (TPR) for visual descriptors only with the positives and avoids collapsing solutions. Furthermore, using none of negatives greatly improves the training efficiency by reducing the scale of the similarity matrix from $\mathcal{O}(n^2)$ to $\mathcal{O}(n)$, and reduces the computation load and memory usage.

To further improve the generalization performance of descriptors, *hard positives* (*i.e.*, corresponding pairs with large-scale transformation) are encouraged as training data to expose novel patterns. However, recent experiments have shown that directly contrastive learning on stronger

[*] Corresponding author

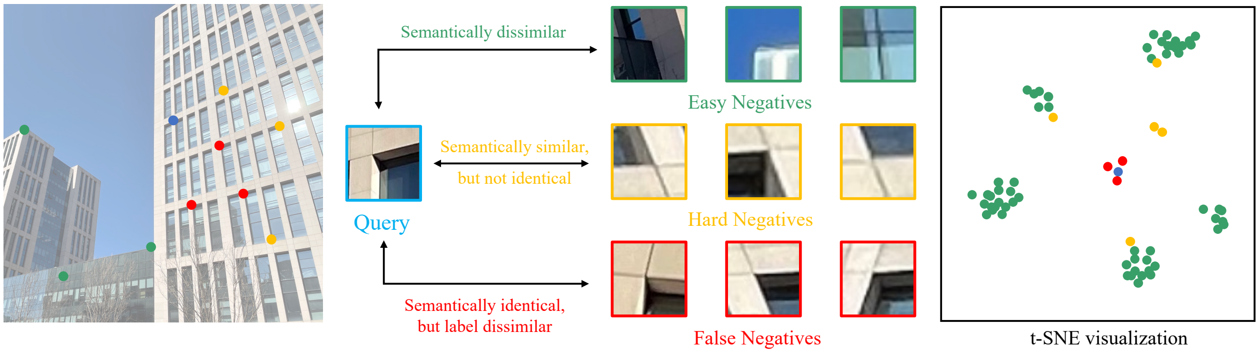

Figure 1. **Illustrative schematic of different negative samples.** *Easy negatives* (green) are easy to distinguish from the *query* (blue) and not sufficient for good performance. *Hard negatives* (orange) are important for better performance and are still semantically dissimilar from the query. *False negatives* (red) are practically impossible to distinguish and semantically identical with the query, which is harmful to the performance. The right figure visualizes the embedding of different samples in latent space.

transformed images can not learn representations effectively [52]. In addition, current contrastive learning methods label all positives with different transformation strength as coarse "1", which prevent learning refined representation. We propose to learn transformation predictive representation with soft positive labels from (0,1] instead of "1" to supervise the learning of local descriptors. Furthermore, we propose a self-supervised curriculum learning module to generate controllable stronger positives with gradually refined soft supervision as the network iterative training.

Finally, our TPR with soft labels is trained on natural images in a fully self-supervised paradigm. Different from previous methods trained on datasets with SfM or camera pose information [20, 29, 39, 49], our training datasets are generally easy to collect and scale up since there is no extra annotation requirement to capture dense correspondences. Experiments show that our self-supervised method outperforms the state-of-the-art on standard image matching benchmarks by noticeable margins and shows excellent generalization capability on multiple downstream tasks (*e.g.*, visual odometry, and localization).

Our contributions to this work are as follows: i) we propose to learn transformation-predictive representations for joint local feature learning, using none of the negative sample pairs and avoiding collapsing solutions. ii) We adopt self-supervised generation learning and curriculum learning to soften the hard positives into continuous soft labels, which can alleviate the false positives and train the model with stronger transformation. iii) The overall pipeline is trained with the self-supervised paradigm, and the training data are computed from random affine transformation and augmentation on natural images.

2. Related Works

Our work focuses on the first step of the image matching pipeline, *i.e.*, detection and description of key-points. This section gives a brief review of key-points detection and description learning together with contrastive representation learning.

Key-points Learning. Different from early hand-crafted key-points, *e.g.*, SIFT [27] and SURF [3], the recent effort has been put into learning key-points through the deep neural networks. Many existent works focus on single module optimization with deep learning, such as interest points detection [42], shape estimation [58] and descriptor representation [45, 47]. However, optimizing one single component may not directly enable the improvement of the entire pipeline [44, 57]. Recent works tend to build end-to-end joint learning frameworks for detecting and describing local features [12, 29, 53]. CNN-based methods simultaneously optimize both the detector and descriptor by sharing most parameters with the help of the encoder-decoder pipeline. Compared with the conventional methods only considering low-level features like corners, edges, or blobs in shallow layers, the encoder-decoder pipeline can make full use of deep CNNs for better representations [11, 25, 39]. To maximize the similarity of corresponding pairs, current cutting-edge methods usually take the Siamese structure to train the model with contrastive learning [54].

Contrastive Learning. Contrastive learning was originated from metric learning and has been widely adopted to supervise the description learning [16, 18, 36, 47]. The critical factor of contrastive learning is to attract the representation of corresponding local descriptions (positive pairs) closer and spread the representations of non-corresponding descriptions (negative pairs) apart. Negative samples are accordingly introduced to keep the uniformity property in order to avoid the representation collapse [13]. Due to expensive manual labeling and the lack of explicit negative signals in most cases, various negative sampling strategies were proposed to improve the computation efficiency and promote the training results [56]. However, the sampling of hard negatives highly depends on the large batch size [6, 7, 54] and memory bank [17], so as to increase the computational load and memory resources usage. Recently, some works have been conducted to optimize the contrastive loss without any negatives and avoid the collapse [5, 9, 15]. But all these works were proposed for the holistic representation of whole images and cannot be adapted to the dense prediction tasks. To address this issue, our method uses an exclusive predictor and stop-gradient operation to avoid collapsing and encourage encoding more transformation-aware representations without negatives, which is the first work ever training key-points with only positives.

3. Method

In this section, we elaborate on the proposed Transformation Predictive Representations (TPR) for learning detection and description of local features. Given a pair of images (I_1, I_2), and the correspondences set $\mathcal{C}$ between them, the key-points learning methods predict the dense 3D description map D and detection heatmap $S \in [0, 1]$ jointly. During the training, the standard Siamese networks are used to deal with the image pairs (I_1, I_2) simultaneously and optimize the parameters with contrastive loss according to $\mathcal{C}$. Our approach also follows this pipeline. However, different from the previous approaches using negatives and hard negative sampling for training, TPR is optimized without any negatives. In this way, our model can reduce the computation complexity and memory usage and improves the training efficiency.

3.1. Preliminary

The architecture in this work is built upon 1) D2-Net [12], and 2) Bootstrap Your Own Latent (BYOL) [15].

D2-Net [12] proposes a describe-and-detect strategy to jointly extract descriptions and detections of local features. Over the last feature maps $y \in \mathbb{R}^{H\times W\times C}$, D2-Net applies channel-wise L2-normalization to obtain the dense feature descriptors, while the feature detections are derived from 1) the local score and 2) the channel-wise score. Specifically, for each location (i, j) in $y^k (k = 1, \ldots, C)$, the final detection score is computed as:

$$s_{ij} = \max_t \left[\frac{\exp(y_{ij}^k)}{\sum_{(i',j')\in\mathcal{N}(i,j)} \exp(y_{i'j'}^k)} \cdot \frac{y_{ij}^k}{\max_t y_{ij}^t} \right], \quad (1)$$

where $\mathcal{N}(i, j)$ is the neighboring pixels around (i, j), *e.g.*, 9 neighbours defined by a 3×3 kernel. D2-Net adopts the triplet margin ranking loss to optimize the descriptions, and is formulated as:

$$\mathcal{L}_{\text{triplet}} = \sum_{c\in\mathcal{C}} \max(0, M + d_p^c - d_n^c), \quad (2)$$

where M is the margin, $\mathcal{C}$ is the set of correspondences, d_p^c and d_n^c represent the distance of positive and selected negative pairs, respectively. Finally, the detection score is used as a weighting term in description loss function.

BYOL [15] is a self-supervised image representation learning approach. BYOL uses two neural networks, referred to as the *online* and *target* networks, both of which interact and learn from each other. The *online* network is trained to predict the *target* network's representation of the same image from an augmented view. The weights of the target network are updated with a slow-moving average of the online network. Different from other contrastive learning methods, BYOL can be trained without any negatives. It hypothesizes that the combination of 1) the addition of a predictor to the online network, and 2) the use of a slow-moving average of the online parameters as the target network encourages encoding more information within the online projection and avoids collapsed solutions.

3.2. Transformation-Predictive Representations

The Siamese-like structure (*i.e.*, the *online* and *target* networks) is used in TPR to learn the representations. From a given representation of the transformed image, referred to as *target*, the TPR trains a new potentially enhanced representation of the original image, referred to as *online*, by predicting the target representation. The online network is defined by a set of weights θ_o and comprises three stages: an online encoder f_o, a projector g_o, and a predictor q. The target network also has the encoder f_t and projector g_t in the same structure but with a different set of weights θ_t. The overall pipeline is described as in Figure 2,

Given the input image $I \in \mathbb{R}^{H\times W\times 3}$, a random augmentation t is performed to produce the cropped view $V \triangleq t\,(I) \in \mathbb{R}^{h\times w\times 3}$. And then, the limited cascaded affine transformation and augmentation T is carried out on the V to produce the transformed view $V' \triangleq T\,(V) \in \mathbb{R}^{h\times w\times 3}$.

From the augmented view V, the online encoder outputs a representation $z \triangleq f_o(V) \in \mathbb{R}^{h\times w\times c}$, which is adopted as the objective encouraged to be transformation-predictive. Rather than predicting representations produced by the online encoder, the target representation $z' \triangleq f_t(V') \in \mathbb{R}^{h\times w\times c}$ is computed by using the target encoder f_t, whose parameters θ_t are an exponential moving average (EMA) of the online encoder parameters θ_o. Without gradient descent, the update algorithm for θ_t is formulated as:

$$\theta_t \leftarrow \tau\theta_t + (1 - \tau)\,\theta_o, \quad (3)$$

where $\tau \in [0, 1)$ is the EMA coefficient.

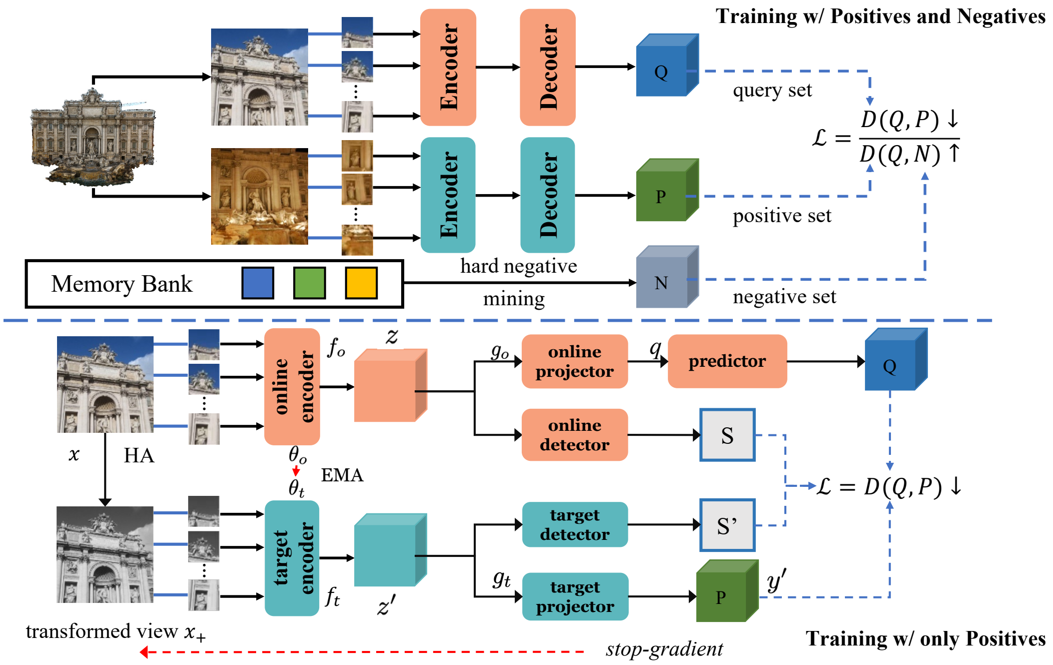

Figure 2. Illustration of our proposed transformation-predictive representations learning. **Top**: traditional contrastive learning descriptors based on both positives and negatives and supervised under SfM models. **Bottom**: our proposed transformation predictive representations learning with only positives and self-supervised loss. In the TPR structure, representations from the online encoder are used in the joint learning of local features task. The target encoder and projection head are defined as an exponential moving average of their online counterparts and are not updated via gradient descent.

Next, the representation z and z' are fed into the online and target detector to extract the detection score s and s' with (1). We further use the online and target projection heads g_o and g_t to map the online and target presentations to a latent embedding space [6], and also apply an additional prediction head q to the online projections to predict the target projections [15]:

$$\begin{aligned} \hat{y} &\triangleq q\left(g_o\left(z\right)\right), \\ y' &\triangleq g_t\left(z'\right). \end{aligned} \tag{4}$$

The target projection head parameters are also given through an EMA of the online projection head parameters by using the same update algorithm as the online and target encoders. Note that the predictor is only applied to online networks so as to make the Siamese architecture asymmetric between the online and target pipeline.

Next, we compute the transformation predictive loss for TPR by summing over cosine similarities between the predicted and target representations. Different from a holistic representation $\mathbb{R}^{1\times 1\times d}$ of the view V and V' [15], our encoder f_o outputs the dense feature maps preserving abundant spatial information with the shape of $h \times w \times c$. The prediction loss is computed on the corresponding location of the dense feature map, *i.e.*, the positive pairs:

$$\begin{aligned} \mathcal{L}_{\text{pred}}^{c} &= \left(1 - \left\langle y_c, y_c' \right\rangle\right) \\ &= \left(1 - \left\langle q\left(g_o\left(z\right)\right)_c, g_t\left(z'\right)_c \right\rangle\right), \end{aligned} \tag{5}$$

where $\langle\cdot\rangle$ denotes the cosine similarity, $\hat{y}_c$ and y_c' is the local representation with the c-th correspondence.

Finally, the detection score is used as a weighting term to formalize the final hard prediction loss function:

$$\mathcal{L}_{\text{hard}} = \frac{1}{|\mathcal{C}|} \sum_{c\in\mathcal{C}} \frac{s_c s_c'}{\sum_{n\in\mathcal{C}} s_n s_n'} \mathcal{L}_{\text{pred}}^{c}. \tag{6}$$

The transformation-predictive representations follow the predictive nature of the objective and use the exponential moving average target network similar to [15]. Compared with the related works on contrastive representation learning, TPR only uses the positive pairs without negative samples. As a result, there is no necessary need on large buffer to emulate large batch sizes of negatives [6, 17] and complex hard negative sampling strategies. Furthermore, training without negatives can help prevent the inconsistent optimization from the *false negatives*. Finally, although we do not use the explicit negative samples to prevent collapse while minimizing $\mathcal{L}_{\text{hard}}$, TPR still can avoid converging to a minimum of this loss with respect to (θ_o, θ_t) (*e.g.*, a collapsed constant representation).

3.3. Learning with Soft Labels

The TPR method uses carefully designed transformations T to generate the view V' from V. Therefore, the views are not deformed aggressively so that they can still be viewed as the same instance, *i.e.*, positives. Stronger deformation could expose the novel patterns of representations in discriminative contrastive learning. However, directly adopting stronger transformations (*e.g.*, with larger rotation angles, more aggressive color jitting, and cutout) may seriously damage the intrinsic semantics of the original image, which will fail to further improve or even degrade the performance [52]. In this paper, we categorize such positive pairs generated from over-strong deformation into the false positives. The false positives are bound to make the training process degraded in performance because the network cannot effectively distinguish the true positives from the false ones when trained with the existing pipeline [39].

To tackle the dilemma, where the strong or weak transformations should not have been used to produce the positives, we propose the optimal soft labels to stabilize the training process instead of forcing the labeling coarse hard "1" for every positive pair. Accordingly, the prediction loss in (5) is modified as follows.

$$\mathcal{L}_{\text{soft}} = \frac{1}{|\mathcal{C}|}\sum_{c\in\mathcal{C}} \frac{s_c s'_c}{\sum_{n\in\mathcal{C}} s_n s'_n}\left(l_c + 1 - \mathcal{L}^c_{\text{pred}}\right), \tag{7}$$

in which l_c represents the computed soft label for the positive pair with c-th correspondence. Therefore, we adopt self-supervised generation learning and curriculum learning to "adjust" the inconsistent and incorrect supervision.

Self-supervised generation learning. We train the network for the first generation following the hard positive label. And then, the network can be assumed to successfully capture most feature distribution of the training data. We propose to utilize the previous generation's similarities to be served as the soft supervision for training the network in a new generation. The generated soft supervisions are gradually refined and sharpened as the network generation progresses. In this way, the system can fully explore the potential of the difficult positives with stronger transformation, and mitigate their label noises with the refined soft confidence. To stabilize the training process, the soft label of the positive pair with the correspondence c in the generation $\omega = 2, \ldots, \Omega$ follows the exponential decay of the transformation strength, which is computed as:

$$l_c = e^{-\left(1-\left\langle y_c^{\omega-1}, {y'}_c^{\omega-1}\right\rangle\right)/\lambda}, \tag{8}$$

where $y^{\omega-1}$ and ${y'}^{\omega-1}$ are generated from the encoder with the last generation parameters θ_ω, and Ω is the total number of generations, and λ is the exponential decay constant.

Curriculum setting for positives generation. The core of curriculum learning is to gradually improve the transformation strength. In particular, we introduce the affine adaption and data augmentation to generate the transformation matrix, which also determines the transformation strength α. We normalize α within $[0, 1]$ according to the maximum limitation of adaption and augmentation parameters, *e.g.*, rotation and translation values. With the curriculum setting, α is randomly sampled from a range with gradually relaxed restrictions of the transformation strength. The soft label of the current batch follows the exponential decay of the transformation strength, which is computed as:

$$l_c = e^{-\alpha\left(1-\left\langle y_c^{\omega-1}, {y'}_c^{\omega-1}\right\rangle\right)/\lambda}. \tag{9}$$

To avoid the over-fitting to the soft labels, we also set a maximum operation on the predicted soft loss as $max(0, l_c + 1 - \mathcal{L}^c_{pred})$. This setting will help the model avoid divergence through the false positives during the early training stage, and encourage the model to explore more robust representations in the later training stage.

In the later stage of training, with the increasing difficulty of positive samples, the proportion of false positive samples, which have different semantics with the online query, will increase. Such false positives will be harmful to the generalization performance of the model [52]. To overcome this problem, we introduce a hyperparameter patience p of early stop. When the soft prediction loss no longer decreases, the value of p will decrease. The training will stop until $p = 0$.

3.4. Implementation Details

In this part, the architectural details of TPR are introduced. Note that at the end of the training, everything but the online encoder f_o is discarded. Thus, the online projection head g_o, the predictor head q, and the target branch will not introduce the extra computational costs in deployment.

Encoder, f_o, maps the input view V into the dense embeddings z. The encoder can be instanced into the CNN version based on VGG (similar as D2-Net [12]) and DCN (similar as ASLFeat [29]), and Transformer version based on tiny Swin Transformer V1 [26] with only the first two stages.

Projection head, g_o, maps the feature embedding into a 128-dim l_2 normalization feature vector for the computation of the prediction loss $\mathcal{L}_{\text{pred}}$. g_o is implemented as a 3-layer MLP (hidden layer 256-dim), with BN and ReLU on every fully-connected layer except the last output layer.

Predictor, q, is only applied to the online networks. The predictor is an MLP with two layers. The dimension of the hidden and output layer is 64 and 128, respectively.

Detection and description head, are adopted on the encoder to simultaneously output the dense description and the detection score map following the D2-Net [12]. Only the locations with higher confidence than 0.7 are selected as the key points. Moreover, a non-maximum suppression (NMS) operation (with a kernel of 5) is applied to remove the key points that are spatially too close.

4. Experiments

In this section, we first present the details of training our TPR. And then we show the performance across several downstream scenarios, including image matching, visual odometry, and visual localization. It's worthy to NOTE that our methods are not trained on all benchmark datasets, which is essential to show generalization performance.

4.1. Experimental Setup

Our TPR method is trained on images from Microsoft COCO dataset[24] and Image Matching Challenge[20], by dropping all the irrelevant annotations. We compute the correspondences between the transformed natural images with the random affine adaption and data augmentation by means of the self-supervised paradigm. For the affine adaption, we uniformly sample the in-plane rotation, shear, translation and scale parameters from [-45°, +45°], [-40°,+40°], [-0.05,+0.05], [0.7,1.4], respectively. For the color jitter, we also uniformly sample the brightness, contrast, saturation, and hue parameters from [0.6,1.4], [0.6,1.4], [0.6,1.4], [-0.2,+0.2], respectively. The additional random data augmentation is employed on the transformed views V and V', including the grayscale conversion and Gaussian blur.

We train models using SGD and AdamW optimizer for CNN and Transformer-based models, respectively. During training, we use a mini-batch size of 16 and an initial learning rate of 0.00017 with exponential decay of 0.9. The hyper-parameter of EMA update τ, exponential decay constant λ, and early stop patience p is set to 0.99, 10, and 1, which are determined with the grid search to reach the optimum. Our method were trained for 50 epochs, which took 30 hours for training with two NVIDIA-A100 GPUs. We randomly crop the 224×224 views for training.

Moreover, we evaluate the local features learned by TPR across several downstream scenarios, including the image matching on HPatches, visual odometry on KITTI, and visual localization on Aachen Day-Night dataset.

4.2. Image Matching

We first evaluate the performance of our method in the image matching tasks, i.e., the most extensive key-points applications.

Datasets. The experiments are taken on HPatches [2] datasets, which include 116 sequences of 6 images with known homography. The datasets are split into two categories according to the illumination and viewpoint changes. The first image in one sequence is taken as reference, and the subsequent images are used to form the pairs with increasing difficulty. By following the previous methods[39], eight high-resolution sequences are excluded.

Method	MMA@3	AUC@2	AUC@5
SIFT [27]	50.1	39.49	49.57
HardNet [32]	62.1	42.61	56.85
LF-Net [35]	53.2	38.74	48.69
SuperPoint [11]	65.7	44.08	59.04
DELF [34]	50.7	44.73	49.70
ContextDesc [28]	63.2	47.23	58.25
Key.Net [22]	72.1	40.87	56.04
R2D2 [39]	72.1	43.35	64.17
DISK [49]	77.2	52.33	69.80
ALIKE [59]	70.5	51.65	69.04
SSL+CAPS [30]	69.0	48.72	62.19
LLF [46]	74.0	52.14	66.81
MTLDesc [50]	78.7	55.02	71.42
PoSFeat [23]	75.34	50.16	69.23
D2-Net [12] (*orig.*)	40.3	19.49	37.78
D2-Net [12] (*our impl.*)	44.5	22.35	43.17
Ours(VGG)	**49.6** ↑ 9.3	**24.46** ↑ 4.97	**47.69** ↑ 9.91
ASLFeat [29] (*orig.*)	72.2	50.10	66.93
ASLFeat [29] (*our impl.*)	74.4	51.83	69.24
Ours(DCN)	**75.5** ↑ 3.2	**52.33** ↑ 2.23	**70.15** ↑ 3.22
Ours(TR)	**79.8**	**57.18**	**73.00**

Table 1. Quantitive results (§4.2) on HPatches.

Evaluation protocols. For a fair comparison, we match the features extracted by each method using the nearest neighbor matcher, accepting only mutual nearest neighbors. A correct match is accordingly considered if its estimated reprojection error is below a given matching threshold. We vary the threshold from 1 pixel to 10 pixels and record the mean matching accuracy (MMA) [31] over all pairs, which represents the ratio of correct matches and possible matches. And then the area is computed under curve (AUC) at 2px and 5px based on the MMA. We report the average scores for overall image pairs.

Comparisons with other methods. We compare our model with the previous state-of-the-art methods on the HPatches dataset. Unless otherwise specified, we present the results either reported on original papers or derived from authors' public implementations with default parameters. Table. 1 lists different methods' results on HPatches in terms of the MMA area under the overall curve (AUC) up to 2px and 5px. Especially in the low-threshold area, **Ours** (TR) shows considerable improvements over previous sota methods. For a fair comparison, we show both the performance from the original paper (*orig.*) and our implementation (*our impl.*). It is observed that TPR gains a +6.69% (44.71% *v.s.* 38.02%) and +2.45% (79.81% *v.s.* 77.36%) MMA promotion than the leading DISK [49] at 1px and 3px threshold in 128-dimension, respectively. To prove the effectiveness of our training pipeline, we also con-

Neg	Sample	+SSGL	+CL	MMA@3	Mem/GB
w/	Random			63.1	112
	Hardest			73.4	140
	Bank			**78.2** ↑ **15.1**	158
	Mixing			77.9	147
w/o	Random			70.1	79
	Random	✓		**77.4** ↑ **7.2**	79
	Random	✓	✓	**79.8** ↑ **9.6**	79

Table 2. **Ablation experiments of the proposed TPR**, where Neg, Sample, MMA@3, SSGL, CL and Mem represent whether training *w/* Negative samples, Sampling strategy, self-supervised generation learning, curriculum learning, mean matching accuracy at 3px threshold, and GPU Memory usage (GB), respectively.

duct experiments based on the D2-Net [12] backbone *i.e.*, **`Ours`** (VGG) and ASLFeat [29] backbone, *i.e.*, **`Ours`** (DCN). Under the same networks, our retrained model gains +9.3% and +3.2% improvements on MMA@3, respectively. The t-SNE visualization of descriptions from D2-Net (*orig.*) and **`Ours`** (VGG) is shown in Figure. 3. More importantly, our method is trained in the self-supervised paradigm, requiring no extra annotations, and is not finetuned on the downstream datasets.

4.3. Ablation Study

We further study the efficacy of our core ideas and essential model designs over HPatches. We train each model from scratch for 50 epochs to perform extensive ablation experiments while keeping other hyper-parameters unchanged.

Ablation on different negative sampling strategies. We adopt the standard Siamese networks with transformer blocks as the "baseline" model. The baseline model is first trained with both positive and negative pairs, according to the triplet loss in (2). We also evaluate different negative sampling strategies including random negative sampling (Random) [10], in-batch hard negative mining (Hardest) [6], hard negative mining with Memory Bank (Bank) [17], and hard negative mixing (Mixing) [21]. And then we train the model using our TPR method with only positives and evaluate the efficacy of the curriculum learning and self-supervised generation learning. As demonstrated in Table 2, our TPR successfully gets convergent with only positives and avoids the collapsing solution, which proves the effectiveness of our designed components in the Siamese networks. Furthermore, the soft labels effectively alleviate the inconsistent optimization from pseudo positives, especially on the training datasets with stronger transformation and augmentation.

Ablation on key components. We also conduct the ablation experiments to evaluate the effectiveness of our core components including training w/o negative samples, self-supervised generation learning (SSGL), and curriculum learning module (CL). The results are also shown in Table 2, which proves the self-supervised curriculum learning with soft labels will improve the contrastive learning performance with only positives.

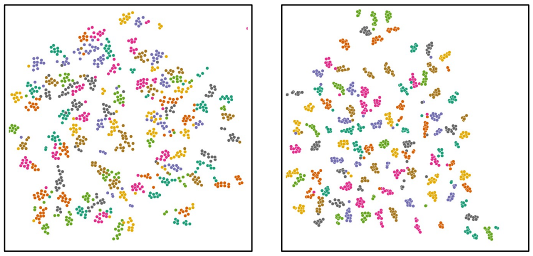

Figure 3. **t-SNE visualization of description from different training methods.** Left: D2-Net [12] (*orig.*), Right: **`Ours`**(VGG).

4.4. Visual Odometry

Visual Odometry (VO) is the task to estimate a robot's position and orientation from visual measurements. The goal of the system is to estimate the 6-DoF pose of the camera at every frame. In this experiment, we evaluate the visual odometry performance with the learned key-points.

Datasets. We adopt the KITTI Odometry Datasets [14] to evaluate the downstream localization performance of our method. We only use the grayscale monocular images with a high resolution of 376 × 1226 as input. To output multiscale key-points, a three-layer image pyramid is constructed with the scale factor of 1.2. It is worth noting that in KITTI, there contains many moving objects and changing camera exposure time, which is challenging for accurate and robust key-points matching.

Evaluation protocols. We take the ORB-SLAM [33] as the benchmark pipeline. We further disable both the local mapping and loop closure thread so as to keep the front-end visual odometry only. To evaluate the key-points fairly, all components in the pipeline are the same except the front-end used key-points. KITTI provides the ground-truth 6-DoF camera poses for sequences 00-10. Since the key-points are adopted in the monocular VO pipeline, we align their poses with ground-truth to recover the scale of 6-DoF poses. Besides, we use the standard evaluation method, i.e., translational root-mean-square error (RMSE) drift (/m), provided along with KITTI dataset. To evaluate the efficiency of the learning-based key-points, we also report the mean running frames per second for the methods.

Results. We report the results of different key-points on different KITTI sequences in Table 3. TPR outperforms the *sota* learning methods DISK [49] on all sequences even under complex traffic conditions. As for the inference time,

Method	FPS	RMSE/m ↓										
		00	01	02	04	05	06	07	08	09	10	*Avg.*
ORB [40]	**20.6**	59.46	610.35	72.68	19.26	238.60	83.46	72.72	66.06	119.21	63.52	140.53
SuperPoint [11]	6.5	162.78	**123.34**	13.52	1.06	6.36	**2.05**	12.15	8.66	8.20	5.10	34.32
D2-Net [12]	8.8	10.44	183.04	105.33	2.29	14.58	2.25	10.72	24.27	29.62	9.61	39.22
R2D2 [39]	7.8	49.62	515.96	60.14	3.90	123.05	62.44	53.84	62.54	73.30	43.32	104.81
SOSNet [48]	6.3	171.67	309.83	10.36	0.47	14.68	4.07	15.35	10.75	3.24	7.67	54.81
DISK [49]	6.5	32.77	149.98	18.67	0.45	5.97	4.38	12.88	32.85	4.33	4.81	26.71
Ours (TR)	6.2	**7.07**	164.39	**9.72**	**0.23**	**3.46**	2.12	**9.99**	**7.42**	**3.10**	**3.72**	**21.12**

Table 3. **Visual odometry localization performance based on different key-points in KITTI datasets.**

Method	*Feat*	Accuracy @ Thresholds (%) ↑		
		0.25m,2°	0.5m,5°	5m,10°
RootSIFT [1]	11K	53.4	62.3	72.3
SuperPoint [11]	7K	68.1	85.9	94.8
D2-Net [12]	14K	67.0	86.4	97.4
R2D2 [39]	10K	70.7	85.3	96.9
ASLFeat [29]	10K	71.2	85.9	96.9
DISK [49]	10K	72.8	86.4	97.4
MTLDesc [50]	7K	74.3	86.9	96.9
Ours (TR)	10K	**74.3**	**89.0**	**98.4**

Table 4. **Performance on Aachen Day-Night Localization datasets.**

our method is slower than the hand-crafted ORB [40]. However, our approach adds no additional overhead compared to existing neural network-based methods. We also report the trajectory visualization in the Supplementary Materials.

4.5. Visual Localization

To further verify the effectiveness of our method confronted with complex tasks, we evaluate it on the task of visual localization, which aims at estimating the camera pose within a given scene using an image sequence. The task was proposed in [41] to evaluate the performance of local features in the context of long-term localization without the need for a specific localization pipeline. All methods are compared on the official evaluation server in a fair manner.

Datasets. We resort to the Aachen Day-Night dataset [41] to demonstrate the effect on visual localization tasks, which contains images from the old inner city of Aachen, Germany. The key challenge in the dataset lies in matching images with extreme day-night changes for 98 queries.

Evaluation protocols. The evaluation is conducted by using *The Visual Localization Benchmark*, which takes a pre-defined visual localization pipeline based on COMLAP [43]. The successful localized images are counted within three error tolerances (0.25m, 2°) / (0.5m, 5°) / (5m, 10°), representing the maximum position error in meters and degrees.

Results. We compare our model with the typical joint detector and descriptor learning methods. Here, all methods are evaluated with the built-in matching strategy (Nearest Neighbors Search) for a fair comparison. As shown in Table 4, TPR with transformer blocks performs surprisingly well even under challenging illumination changes, about the estimated pose, which demonstrates the effectiveness of self-supervised learning with only positives on the natural dataset.

4.6. Limitations

While our approach does not require any additional annotation, e.g., Structure-from-Motion models or optical flow, to produce the corresponding real image pairs. It still needs to receive pairs through homography adaption or image augmentation. Furthermore, despite getting better performance without using negative samples, TPR local features still fail in classical challenging cases such as untextured areas, especially in the absence of matching priors.

5. Conclusion

In this paper, we propose to learn the transformation-predictive representations for the joint detection and description of local features. The model is trained by using none of the negatives and avoids the collapsing solution, which greatly improves the training efficiency. Self-supervised generation learning and curriculum learning are designed to soften the hard positives into continuous soft labels, which can train the model with stronger augmentation. We solve the label noise from false positives and negatives and further improve the performance of local features in the self-supervised paradigm. Experiments show that TPR significantly outperforms state-of-the-art methods.

Acknowledgments

This work was supported by the Science and Technology on Space Intelligent Control Laboratory under Grant HTKJ2021KL502019.

References

[1] Relja Arandjelović and Andrew Zisserman. Three things everyone should know to improve object retrieval. In *CVPR*, 2012. 8

[2] Vassileios Balntas, Karel Lenc, Andrea Vedaldi, and Krystian Mikolajczyk. Hpatches: A benchmark and evaluation of handcrafted and learned local descriptors. In *CVPR*, 2017. 6

[3] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *ECCV*, 2006. 2

[4] Tiffany Tianhui Cai, Jonathan Frankle, David J Schwab, and Ari S Morcos. Are all negatives created equal in contrastive instance discrimination? *arXiv preprint arXiv:2010.06682*, 2020. 1

[5] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *NeurIPS*, 2020. 1, 3

[6] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020. 1, 3, 4, 7

[7] Ting Chen, Yizhou Sun, Yue Shi, and Liangjie Hong. On sampling strategies for neural network-based collaborative filtering. In *ACM SIGKDD*, 2017. 3

[8] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv:2003.04297*, 2020. 1

[9] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *CVPR*, 2021. 1, 3

[10] Peng Cui, Shaowei Liu, and Wenwu Zhu. General knowledge embedded image representation learning. *IEEE TMM*, 2017. 7

[11] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *CVPR*, 2018. 2, 6, 8

[12] Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-net: A trainable cnn for joint detection and description of local features. In *CVPR*, 2019. 1, 2, 3, 5, 6, 7, 8

[13] Tianyu Gao, Xingcheng Yao, and Danqi Chen. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*, 2021. 1, 2

[14] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012. 7

[15] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *NeurIPS*, 2020. 1, 3, 4

[16] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *AISTATS*, 2010. 2

[17] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020. 1, 3, 4, 7

[18] Kun He, Yan Lu, and Stan Sclaroff. Local descriptors optimized for average precision. In *CVPR*, 2018. 2

[19] Jared Heinly, Johannes L Schonberger, Enrique Dunn, and Jan-Michael Frahm. Reconstructing the world* in six days*(as captured by the yahoo 100 million image dataset). In *CVPR*, 2015. 1

[20] Yuhe Jin, Dmytro Mishkin, Anastasiia Mishchuk, Jiri Matas, Pascal Fua, Kwang Moo Yi, and Eduard Trulls. Image matching across wide baselines: From paper to practice. *IJCV*, 2021. 2, 6

[21] Yannis Kalantidis, Mert Bulent Sariyildiz, Noe Pion, Philippe Weinzaepfel, and Diane Larlus. Hard negative mixing for contrastive learning. *NeurIPS*, 2020. 1, 7

[22] Axel Barroso Laguna and Krystian Mikolajczyk. Key. net: Keypoint detection by handcrafted and learned cnn filters revisited. *IEEE TPAMI*, 2022. 6

[23] K. Li, LongguangWang, L. Liu, Q. Ran, K. Xu, and Y. Guo. Decoupling makes weakly supervised local feature better. In *CVPR*, 2022. 6

[24] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 6

[25] Dongfang Liu, Yiming Cui, Liqi Yan, Christos Mousas, Baijian Yang, and Yingjie Chen. Densernet: Weakly supervised visual localization using multi-scale feature aggregation. In *AAAI*, 2021. 2

[26] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *CVPR*, 2021. 5

[27] David G Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 2004. 2, 6

[28] Zixin Luo, Tianwei Shen, Lei Zhou, Jiahui Zhang, Yao Yao, Shiwei Li, Tian Fang, and Long Quan. Contextdesc: Local descriptor augmentation with cross-modality context. In *CVPR*, 2019. 6

[29] Zixin Luo, Lei Zhou, Xuyang Bai, Hongkai Chen, Jiahui Zhang, Yao Yao, Shiwei Li, Tian Fang, and Long Quan. Aslfeat: Learning local features of accurate shape and localization. In *CVPR*, 2020. 1, 2, 5, 6, 7, 8

[30] Iaroslav Melekhov, Zakaria Laskar, Xiaotian Li, Shuzhe Wang, and Juho Kannala. Digging into self-supervised learning of feature descriptors. In *IEEE 3DV*, 2021. 6

[31] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *IEEE TPAMI*, 2005. 6

[32] Anastasiia Mishchuk, Dmytro Mishkin, Filip Radenovic, and Jiri Matas. Working hard to know your neighbor's margins: Local descriptor learning loss. In *NeurIPS*, 2017. 6

[33] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 2015. 1, 7

[34] Hyeonwoo Noh, Andre Araujo, Jack Sim, Tobias Weyand, and Bohyung Han. Large-scale image retrieval with attentive deep local features. In *ICCV*, 2017. 6

[35] Yuki Ono, Eduard Trulls, Pascal Fua, and Kwang Moo Yi. Lf-net: learning local features from images. In *NeurIPS*, 2018. 6

[36] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv:1807.03748*, 2018. 2

[37] Luc Oth, Paul Furgale, Laurent Kneip, and Roland Siegwart. Rolling shutter camera calibration. In *CVPR*, pages 1360–1367, 2013. 1

[38] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *CVPR*, 2007. 1

[39] Jerome Revaud, Philippe Weinzaepfel, César De Souza, Noe Pion, Gabriela Csurka, Yohann Cabon, and Martin Humenberger. R2d2: Repeatable and reliable detector and descriptor. In *NeurIPS*, 2019. 1, 2, 5, 6, 8

[40] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *ICCV*, 2011. 8

[41] Torsten Sattler, Will Maddern, Carl Toft, Akihiko Torii, Lars Hammarstrand, Erik Stenborg, Daniel Safari, Masatoshi Okutomi, Marc Pollefeys, Josef Sivic, et al. Benchmarking 6dof outdoor visual localization in changing conditions. In *CVPR*, 2018. 8

[42] Nikolay Savinov, Akihito Seki, Lubor Ladicky, Torsten Sattler, and Marc Pollefeys. Quad-networks: unsupervised learning to rank for interest point detection. In *CVPR*, 2017. 2

[43] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 8

[44] Johannes L Schonberger, Hans Hardmeier, Torsten Sattler, and Marc Pollefeys. Comparative evaluation of hand-crafted and learned local features. In *CVPR*, 2017. 2

[45] Edgar Simo-Serra, Eduard Trulls, Luis Ferraz, Iasonas Kokkinos, Pascal Fua, and Francesc Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. In *ICCV*, 2015. 2

[46] Suwichaya Suwanwimolkul, Satoshi Komorita, and Kazuyuki Tasaka. Learning of low-level feature keypoints for accurate and robust detection. In *WACV*, 2021. 6

[47] Yurun Tian, Bin Fan, and Fuchao Wu. L2-net: Deep learning of discriminative patch descriptor in euclidean space. In *CVPR*, 2017. 2

[48] Yurun Tian, Xin Yu, Bin Fan, Fuchao Wu, Huub Heijnen, and Vassileios Balntas. Sosnet: Second order similarity regularization for local descriptor learning. In *CVPR*, 2019. 8

[49] Michał Tyszkiewicz, Pascal Fua, and Eduard Trulls. Disk: Learning local features with policy gradient. *NeurIPS*, 2020. 2, 6, 7, 8

[50] Changwei Wang, Rongtao Xu, Yuyang Zhang, Shibiao Xu, Weiliang Meng, Bin Fan, and Xiaopeng Zhang. Mtldesc: Looking wider to describe better. *AAAI*, 2022. 6, 8

[51] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *ICML*, 2020. 1

[52] Xiao Wang and Guo-Jun Qi. Contrastive learning with stronger augmentations. *arXiv preprint arXiv:2104.07713*, 2021. 2, 5

[53] Zihao Wang, Xueyi Li, and Zhen Li. Local representation is not enough: Soft point-wise transformer for descriptor and detector of local features. In *IJCAI*, 2021. 2

[54] Zihao Wang, Zhen Li, Xueyi Li, Wenjie Chen, and Xiangdong Liu. Graph-based contrastive learning for description and detection of local features. *IEEE TNNLS*, 2022. 2, 3

[55] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *CVPR*, 2018. 1

[56] Lanling Xu, Jianxun Lian, Wayne Xin Zhao, Ming Gong, Linjun Shou, Daxin Jiang, Xing Xie, and Ji-Rong Wen. Negative sampling for contrastive representation learning: A review. *arXiv preprint arXiv:2206.00212*, 2022. 3

[57] Kwang Moo Yi, Eduard Trulls, Vincent Lepetit, and Pascal Fua. Lift: Learned invariant feature transform. In *ECCV*, 2016. 2

[58] Kwang Moo Yi, Yannick Verdie, Pascal Fua, and Vincent Lepetit. Learning to assign orientations to feature points. In *CVPR*, 2016. 2

[59] Xiaoming Zhao, Xingming Wu, Jinyu Miao, Weihai Chen, Peter CY Chen, and Zhengguo Li. Alike: Accurate and lightweight keypoint detection and descriptor extraction. *IEEE TMM*, 2022. 6