LEARNING LOCAL LINE DESCRIPTORS WITH FCNNS AND TRANSFER LEARNING BY MINIMIZING TRIPLET MARGIN LOSS

Zhanqiang HUO[1], Miaomiao FU[2], Fen LUO[3]*, Yujie LIU[4]

In this paper, it is demonstrated how to learn line feature descriptors for line matching from image data, which is very important for many image processing and computer vision applications. In order to encode these functions, the FCNNs (Fully Convolutional Neural Networks) based models are developed. The main contributions of this method can be summarized form four aspects: (1) it describes the line features with mean patches and standard deviation patches; this idea comes from MSLD (mean–standard deviation line descriptor); (2) three FCNNs based models are introduced, and the central-surrounding to data augmentation is used before training; (3) the parameters of the pre-trained L2-Net are transferred to the three models to solve the problem of insufficient amount of training data; (4) the triplet margin loss is used to optimize these three models. The proposed method is verified by the experiments on the Oxford dataset and other datasets, and the experimental results show that the line feature descriptors obtained by our three models have better robustness under the conditions of viewpoint change, rotation change, blur change, and scale change and many others.

Keywords: Fully Convolutional Neural Networks; transfer learning; line feature descriptor

1. Introduction

The description of local image features is one of the most basic tasks in the image processing and computer vision fields. It is often used as a subroutine and plays a vital role in various computer vision tasks, including image registration [1], image stitching [2], and 3D reconstruction [3]. In the past two decades, various methods for the local image feature description have been proposed. The most popular handcrafted feature descriptor is the Scale Invariant Feature Transform (SIFT) feature descriptor, which has the advantages of ensuring the

[1] Prof., School of Computer Science and Technique, Henan Polytechnic University, Jiaozuo 454003, China
[2] PG., School of Computer Science and Technique, Henan Polytechnic University, Jiaozuo 454003, China
[3]* Assis., School of Computer Science and Technique, Henan Polytechnic University, Jiaozuo 454003, China, Corresponding author: luofenjsj@hpu.edu.cn
[4] PG., School of Computer Science and Technique, Henan Polytechnic University, Jiaozuo 454003, China

invariance of the scale variety of local features and strong distinguishability [4]. Dong and Soattos proposed to accumulate the SIFT results of multiple different scales to obtain the Domain-Size-Pooling SIFT (DSP-SIFT) and explained its superior performance from the perspective of signal processing [5].

In recent years, due to the successful application of deep learning in many fields, the research focus has gradually been turned from the handcrafted descriptors to the learning-based descriptors. For instance, Zagoruyko *et al*. proposed many neural networks based models, including Siamese networks, two-channel networks (2chnet); these models have excellent performances [6] in invariance and distinguishability. Tian *et al*. proposed a Convolutional Neural Network (CNN) model named the L2-Net that uses the progressive sampling strategy and the loss function composed of three error terms to learn the point feature descriptors [7]. Based on the L2-Net architecture and inspired by the SIFT matching criteria, Mishchuk *et al*. proposed a descriptor named the HardNet, where the triple loss was applied to the L2-Net architecture to learn point feature descriptors, thus further improving the matching performance of the L2-Net [8].

In the literature, many line feature descriptors for line feature matching, including the mean-standard deviation line descriptor (MSLD) for line matching proposed by Wang *et al*., have been reported. The MSLD provides a robust descriptor for line matching [9]. In order to overcome the problems of segmentation fragmentation and geometric variation, Zhang *et al*. proposed a line feature descriptor called the Line Band Descriptor (LBD) [10], which has good matching performance on various image changes. Liu *et al*. proposed an intensity order curve descriptor (IOCD). The IOCD partitions the sub-region based on the intensity of pixels in the neighborhood and achieves good distinguishability under the conditions of deformation and complex illumination changes [11].

Although the above-mentioned descriptors have good performances, the development of high-quality local line feature descriptors is still challenging due to the factors that affect the final image appearance and problems with the line itself. The factors affecting the image appearance include changes in the viewpoint, the overall illumination of the scene, rotation, scale, and occlusion. The problems with the line include the uncertainty of the endpoint of the line, unavailability of a strong disambiguating geometric constraint, a lack of rich textures in a local neighborhood of the line [9]. Therefore, compared with the point feature descriptor, the line feature descriptor develops slowly and still stays in the handcrafted stage. Therefore, how to use the existing dataset HPUPatches, a dataset built by ourselves, to learn line feature description functions automatically is meaningful work.

Aiming at solving the above-mentioned problems, this paper proposes to learn local line feature descriptors using the FCNNs and transfer learning while minimizing the triplet margin loss. The main objective is to use the FCNNs and

transfer learning to learn the line feature description function directly from the pre-processed image patch without using any handcrafted features. Inspired by the L2-Net and transfer learning [12], a deep fully-convolutional neural network architecture is combined with transfer learning to represent line features. In this work, three fully convolutional neural networks for line feature descriptor learning are studied. These networks are trained with pre-processed image patches.

The main contributions of this paper are as follows: (i) the line feature description function is learned directly from the image patch data, and various image transformations are implicitly considered; (ii) three types of network architectures suitable for this function learning while improving the matching performance of line descriptors are studied; (iii) the line descriptors applied to the line matching problem using the benchmark dataset, and the results indicate that it performs better than the existing handcrafted descriptors and further enhances the descriptive power of the line feature descriptor.

2. Line dataset construction and network architectures

2.1 Line dataset construction

A line dataset denoted as HPUPatches dataset was constructed, where HPU stands for Henan Polytechnic University. The dataset consisted of two subsets, HPUPatches-train and HPUPatches-test. The HPUPatches-train and HPUPatches-test consisted of approximately 180,000 64×128 image patches and 90,000 64×128 image patches, respectively. The following seven image changes were involved in each subset: illumination, blur, viewpoint, rotation, JPEG compression, noise, and scale. The four concrete steps of constructing the dataset were as follows.

Step 1: By using mobile phone shooting and network downloading, a total of 3400 pairs of images with different changes in the same scene were obtained, involving seven transformations: illumination, blur, viewpoint, rotation, JPEG compression, noise, and scale.

Step 2: Canny edge detection operator was used for image edge detection; the points with the curvature greater than 0.8 were removed; finally, the lines with the number of pixels less than 20 were discarded to obtain extracted lines.

Step 3: A positive matching line pair was obtained. The specific method was as follows. For any image pair, the matching line pair in the image pair was obtained by the line matching technique MSLD, and the matching error was manually eliminated to obtain the positive matching line pair set $S(L,L')$ in the image pair, which was expressed as:

$$S(L,L') = \{(L_j, L_j'), j = 1,2,\dots,N_L\} \quad (1)$$

where L_j denoted the *j*th line in the first image of an image pair, L_j' denoted the *j*th line in the second image of an image pair that matches L_j positive, and N_L denoted the number of matched line pairs.

Step 4: The line feature patches were determined corresponding to the line. The specific method was as follows. For any line L composed of $Num(L)$ points in the set of positive matching line pairs obtained in Step 3, it was noted that any pixel point on L was P_k, $k = 1,2,\ldots,Num(L)$. This step included Steps 4.1–4.3.

Step 4.1: Get the support area for each point on the line. The specific method was as follows. A square region having a length of 64 along the direction of line L and the direction perpendicular to line L, which was centered on P_k, was defined as a support region of a point P_k. The matrix of the brightness values of the point P_k support region was denoted as $I(P_k)$.

Step 4.2: Get the mean patch and standard deviation patch of the line feature. The specific method was as follows: the mean matrix $M(L)$ and the standard deviation matrix $STD(L)$ of the straight line L were calculated according to the support regions of the points acquired in Step 4.1 as follows:

$$M(L) = \frac{I(P_1)+I(P_2)+\cdots+I(P_{Num(L)})}{Num(L)} \tag{2}$$

$$STD(L) = \frac{(I(P_1)-M(L))^2+(I(P_2)-M(L))^2+\cdots+(I(P_{Num(L)})-M(L))^2}{Num(L)} \tag{3}$$

Step 4.3: Get line feature patches. Concatenate the mean patch and standard deviation patch of the line L to obtain the line feature patch $A(L)$ of the size 64×128 pixels by:

$$A(L) = \left[\frac{M(L)}{max(M(L))} \times 255, \frac{STD(L)}{max(STD(L))} \times 255\right] \tag{4}$$

2.2 Network architectures

A neural network can process image patches in many ways, and multi-resolution information can improve the matching ability of descriptors. Therefore, an image patch with a size of 64×128 pixels was subjected to the downsampling and center cropping to obtain the surrounding low-resolution image patch and the central, high-resolution image patch, which were used as an input of the three network models studied in this work. The image patch with a size of 64×128 pixels included a mean patch and a standard deviation patch, of which each had a size of 64×64 pixels and was calculated from the point patches that made up the line. The obtained image patches were used to explore and test three neural network architectures, namely a Central-surround Siamese network (CS S-Net), a

Central-surround Pseudo- Siamese network (CS PS-Net), and a Central-surround 2-Channel network (CS 2-Channel-Net).

2.2.1 CS S-Net

In the CS S-Net, there are two branch structures that share the same architecture and the same parameter set. The CS S-Net architecture is shown in Fig. 1. The two branching structures of the network are processed at two different resolutions, surround low-resolution and central high-resolution. More specifically; and, a 256-dimensional feature vector is obtained by processing the input by seven convolution layers. In the second branch structure (central high-resolution), the center of the mean patch and the standard deviation patch are used as an input to obtain image patch of 32×64 pixels; and then obtain a 256-dimensional feature vector is obtained by processing the input by seven convolution layers. Finally, the output feature vectors of the two branches are merged into the final output of the entire network.

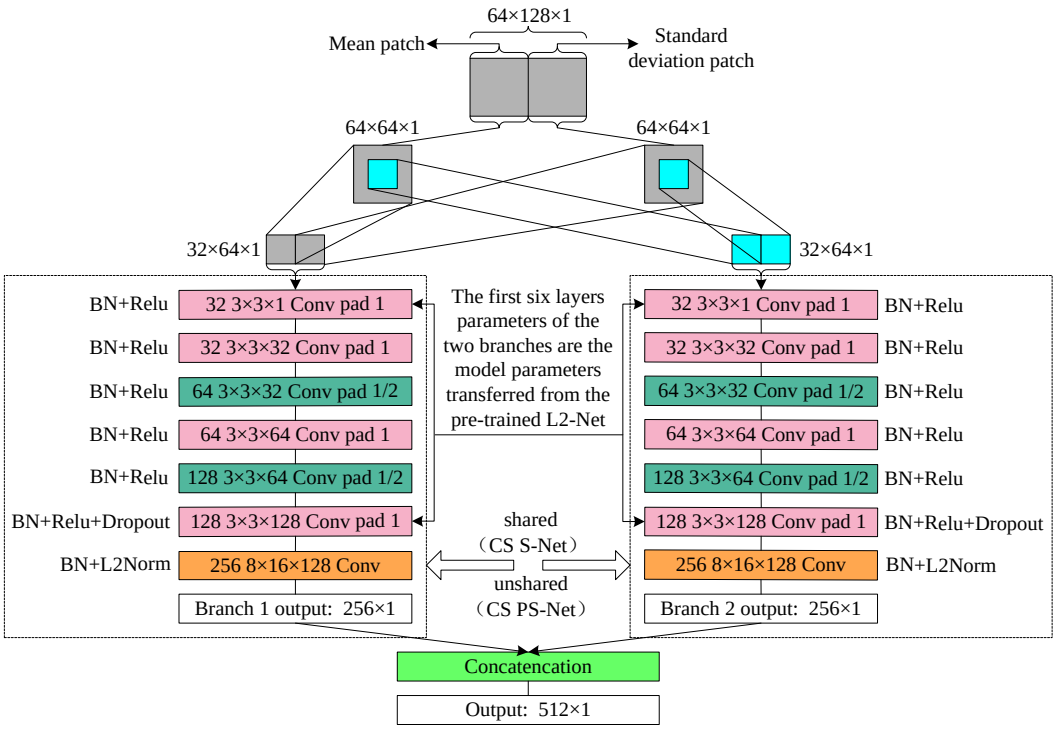

Fig. 1. The architectures of the CS S-Net and CS PS-Net networks. The only difference between the two networks is that the two branches of the latter do not share parameters.

These two branch structures have the same convolutional layers as the L2-Net [7] except for the last convolutional layer. Since the different inputs and the same desired outputs, the convolution kernel size of the last convolution layer is modified from 8×8 to 8×16, and the number of convolution kernels is modified

from 128 to 256. In addition, the dropout regularization with the loss rate of 0.3 is applied before the last layer. Due to the insufficient number of training samples in the existing dataset, the transfer learning is used to set the initial values of the first six layers of the two branches to the model parameter values of the L2-Net pre-trained with the large dataset Liberty [13]; the Liberty contains approximately 450,000 image patches with the size of 64×64 pixels. The last layer of weight tensor is initialized to an orthogonal matrix [14] with the gain equal of 0.6, and the bias tensor is constant and set to 0.01. The CS S-Net converts the input two 32×64 pixels image patches into 512-dimensional descriptors.

2.2.2 CS PS-Net

From a complexity perspective, the CS PS-Net is between CS S-Net and CS 2-Channel-Net. As shown in Fig. 1, it has the structure of the CS S-Net, but the network parameters of the two branches are not shared, which increases the number of parameters that can be adjusted during the training and provides more flexibility than the CS S-Net, but is less flexible than the CS 2-Channel-Net.

2.2.3 CS 2-Channel-Net

Each branch of the CS S-Net and CS PS-Net performs a feature extraction process, and the feature vectors extracted by the two branches are concatenated into the final network output. The CS 2-Channel-Net skips the process of two branches conducting the feature extraction but regards the two 32×64 pixels image patches obtained by the method described in Section 2.2.1 as a two-channel image patch having the size of 32×64×2. Moreover, the patch of size 32×64×2 is fed to the first layer of the convolutional layer. The structure of the CS 2-Channel-Net is shown in Fig. 2, where it can be seen that it converts the input 32×64×2 image patch into a 256-dimensional feature vector by seven convolution layers. Compared to the two networks described above, the CS 2-Channel-Net provides greater flexibility since it processes two patches simultaneously.

3. Triplet margin loss and model training

Transfer learning and supervised learning are used to train the three models. A triplet margin loss based on the distance between the minimized match descriptor and the closest non-matching descriptor is used, which leads to the learning objective function L, which is expressed as:

$$D = \begin{bmatrix} d(a_1, b_1) & d(a_1, b_2) & \dots & d(a_1, b_n) \\ d(a_2, b_1) & d(a_2, b_2) & \dots & d(a_2, b_n) \\ & \dots & & \\ d(a_n, b_1) & d(a_n, b_2) & \dots & d(a_n, b_n) \end{bmatrix}_{n \times n} \tag{5}$$

$$d(a_i, b_j) = \sqrt{2 - 2a_i b_j}\,, i, j = 1,2, \dots, n \tag{6}$$

$$L = \frac{1}{n}\sum_{i=1}^{n} max\left(0{,}1 + d(a_i, b_i) - min\left(d(a_i, b_{j_{min}}), d(a_{k_{min}}, b_i)\right)\right) \quad (7)$$

where n denotes the batch size; a_i and b_i denote the feature description vectors that represent the network output for the network input consisted of two matched image patches; $b_{j_{min}}$ denotes the closest a_i but non-matching line descriptor, and $a_{k_{min}}$ denotes the closest b_i but non-matching line descriptor, while $d(a_i, b_i)$ denotes the distance between two descriptors.

$$j_{min} = argmin\, d(a_i, b_j), j = 1{,}2, \ldots, n, j \neq i \quad (8)$$

$$k_{min} = argmin\, d(a_k, b_i), k = 1{,}2, \ldots, n, k \neq i \quad (9)$$

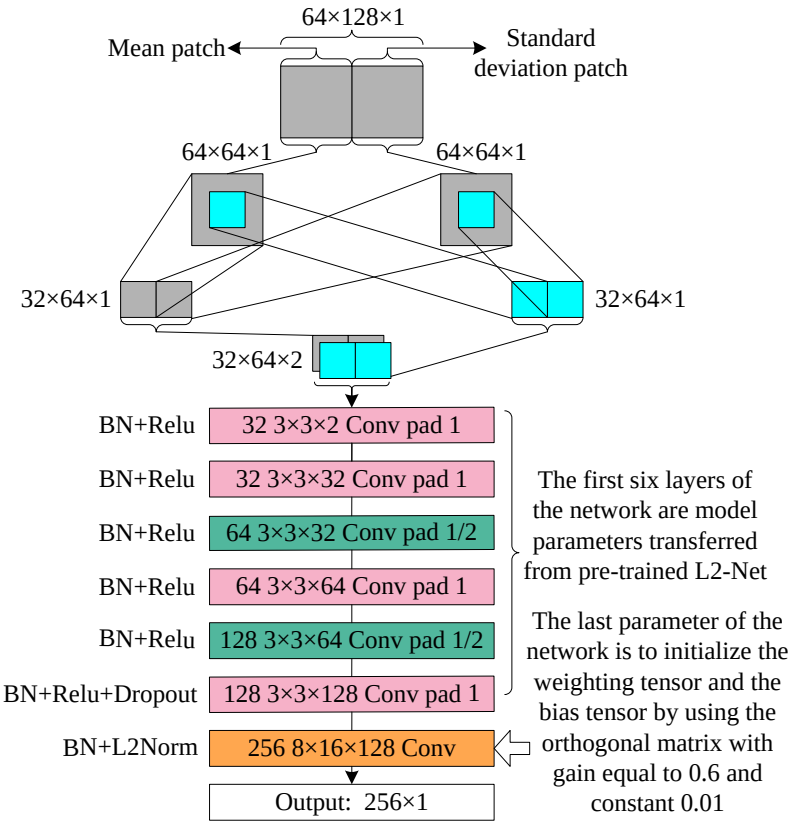

Fig. 2. The structure of the CS 2-Channel-Net network.

The three networks were trained using an SGD with an initial learning rate of 10, a momentum of 0.9, and a weight attenuation of 0.0001. Training was done in mini-batches with a size of 512. Since the construction of a large dataset for the line feature network training requires much manpower and financial resources, and the combination of transfer learning and convolutional neural networks has achieved great successes in computer vision applications, such as image recognition [15] and target tracking [16]. In this work, the transfer learning was

introduced in the fully convolutional neural networks, which means that the first six layer's parameters of the three networks were initialized to the model parameter values of the L2-Net pre-trained by the large dataset Liberty. The last layer of weight tensor was initialized to an orthogonal matrix with a gain of 0.6, and the bias tensor was set to 0.01. The training samples in the existing datasets are insufficient, and there are also over-fitting problems caused by training from scratch. Moreover, for CNN, the first few layers learn the low-level general features of the image, which is suitable for most visual tasks, while several latter layers learn the high-level features for specific tasks [17]. Therefore, a fine-tuning strategy was chosen to train all network models.

The size of the constructed dataset allowed loading all image patches into GPU (Graphic Processing Unit) memory and quickly acquiring the image patches during the training. The RTX 2080Ti GPU in Pytorch [18] was used to train the three networks; the training of all the models converged after ten epochs, so the training process lasted about 3 to 4 hours.

4. Experimental evaluation

The trained models were applied to a variety of datasets. The experimental results obtained by the three network models were compared with the results of the state-of-the-art methods.

In order to evaluate the performance of the line feature descriptors proposed in this paper, FPR95 and mAP (mean Average Precision) evaluation indicators were used. The FPR95 is the False Positive Rate (FPR) at 95% recall, and it is computed as:

$$FPR = \frac{FP}{FP+TN} \tag{10}$$

where *FP* (False Positive) indicates that the unmatched sample is predicted to be matched, and *TN* (True Negative) indicates that the unmatched sample is correctly predicted as unmatched.

In image matching, mAP is used as a performance evaluation indicator. The AP (Average Precision) of a matching category on an image pair is calculated as:

$$AP = \frac{\sum_{j=1}^{np} Precision_j}{np} \tag{11}$$

where *np* represents the total number of retrieved positively matched line pairs, and *Precision* denotes the ratio of the number of the retrieved positively matched line pairs to the total number of retrieved line pairs. The mAP is then calculated as:

$$mAP = \frac{\sum_{i=1}^{m} AP_i}{m} \tag{12}$$

where *m* denotes the total number of image pairs in the test set.

4.1 HPUPatches dataset

The three network models were first trained using a subset of the HPUPatches-train, and then the performances of the three models were tested using a subset of the HPUPatches-test. The results of the CS S-Net, CS PS-Net, and CS 2-Channel-Net models regarding the performance indicator FPR95 are presented in Fig. 3. As presented in Fig. 3, the CS 2-Channel-Net achieved the best performance. This demonstrates the importance of the direct combination of the information from the two patches from the first network layer, and the importance of the multi-resolution information on the line description task. The results obtained by the CS PS-Net were comparable to those of the CS S-Net.

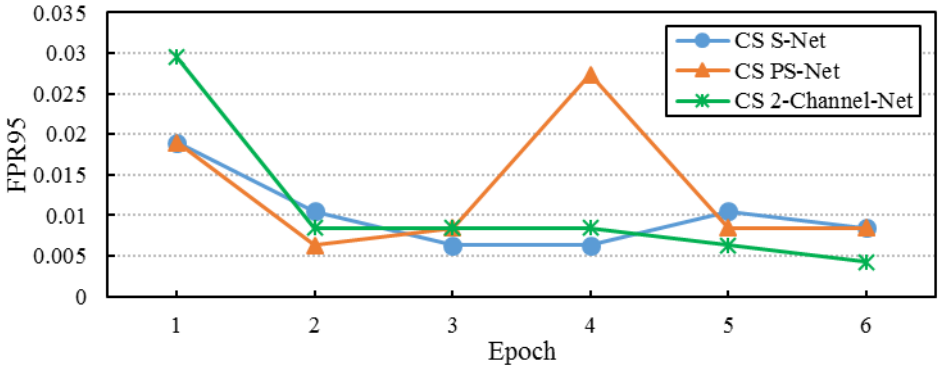

Fig. 3. Experimental results of the CS S-Net, CS PS-Net, and CS 2-Channel-Net models regarding the performance indicator FPR95.

4.2 Oxford dataset

The Oxford dataset [19], which is a standard dataset, was used to evaluate the generalization capabilities of the three network models. The CS S-Net, CS PS-Net, and CS 2-Channel-Net were evaluated on seven sets of image sequences: Bikes (blur change), Graffiti (viewpoint and rotation combination change), Leuven (illumination change), Ubc (JPEG compression change), Wall (viewpoint change), Trees (blur change), and Boat (rotation and scale combination change). There are six images in each group of image sequences, and the last five images in each group had different degrees of changes compared with the first image. The performance of the CS S-Net, CS PS-Net, CS 2-Channel-Net, and the existing advanced handcrafted methods were evaluated by matching the last five images in each set of image sequences to the first image.

The handcrafted descriptors used in the comparison were the MSLD [9] and the IOCD [11]. The comparison results regarding the mAP are presented in Fig. 4. For a fair comparison, all methods were adjusted to have the same number of feature lines. As presented in Fig. 4, the CS S-Net, CS PS-Net, and CS 2-Channel-Net achieved the best matching performance; they performed

significantly better than the MSLD and IOCD especially under the conditions of changes in the blur (Bikes and Trees), viewpoint (Graffiti and Wall) and rotation and scale combination (Boat). The results of the total number of positive samples recalled in seven sets of image sequences are presented in Table 1, where it can be seen that the CS S-Net, CS PS-Net, and CS 2-Channel-Net achieved a significant increase in the total number of positive samples recalled compared to the other handcrafted descriptors.

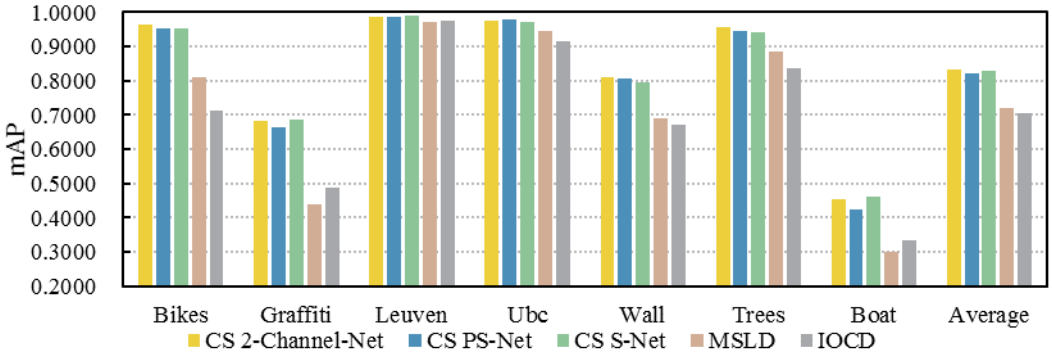

Fig. 4. Experimental results of the mAP on the Oxford dataset.

Table 1

Results of the total number of positive samples on the Oxford dataset

	Bikes	Graffiti	Leuven	Ubc	Wall	Trees	Boat	All
CS PS-Net	416	248	516	357	58	99	256	1950
CS S-Net	416	241	512	356	58	99	256	1938
CS 2-Channel-Net	415	238	517	357	58	99	254	1938
MSLD	364	171	480	329	52	79	184	1659
IOCD	352	234	501	320	56	85	219	1767

4.3 Other dataset

In order to further prove the generalization capabilities of the CS S-Net, CS PS-Net and CS 2-Channel-Net networks, a dataset consisting of four sets of image sequences from the traditional handcrafted line feature description article [10], which includes changes in the viewpoint, scale, low texture, and occlusion, was used. The comparison results of the mAP are presented in Fig. 5, where it can be seen that: 1) in terms of viewpoint and scale changes, the performances of the proposed models were significantly better than those of the MSLD and IOCD; 2) when the dataset HPUPatches-train did not contain occlusion and low-texture changes, the proposed models were superior to the MSLD and IOCD in the case of low-texture changes; and under the condition of occlusion change, the CS S-Net, CS PS-Net, and CS 2-Channel-Net were superior to the MSLD and comparable to the IOCD. The line matching results of the proposed descriptors for

four pairs of images at NNDR of 0.8 are presented in Fig. 6. The results presented in Fig. 6 further prove the effectiveness of the proposed descriptors.

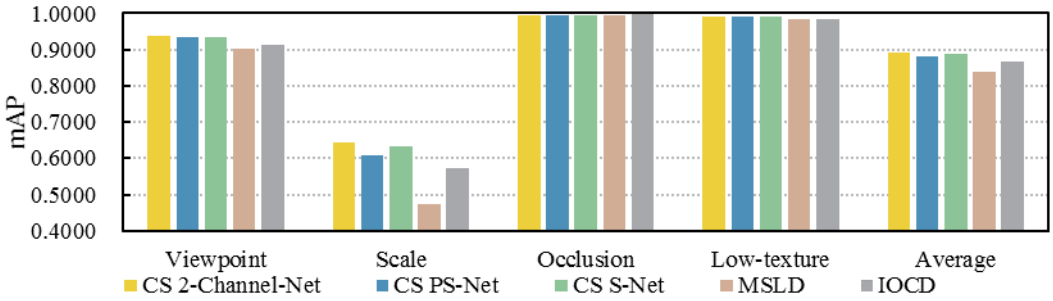

Fig. 5. Results of different models for various image changes in the dataset provided in the traditional handcrafted line feature description article [10].

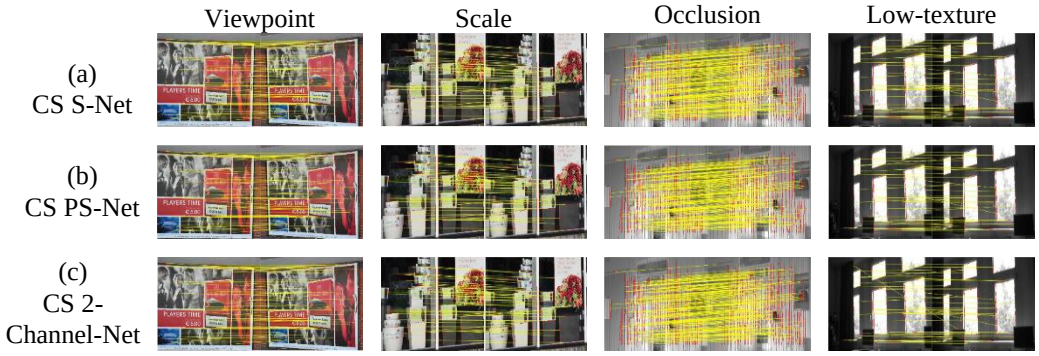

Fig. 6. The matching results of the CS S-Net, CS PS-Net, and CS 2-Channel-Net descriptors.

5. Conclusion

In this paper, it is shown how the line feature description functions can be leaned directly from image patches, which are encoded in the form of a fully convolutional model. Three neural network architectures that are suitable for this task are studied. Due to the lack of the existing training samples, transfer learning is adopted to develop three neural network architectures suitable for line feature description tasks by fine-tuning the L2-Net model pre-trained on the Liberty dataset. Moreover, it is demonstrated that the developed models outperform two popular handcrafted descriptors on several standard datasets, especially in the presence of significant changes in the blur, viewpoint, rotation, and scale. Among the three developed networks, the CS 2-Channel-Net network architecture achieved the best results. However, these three network architectures are sensitive to occlusion changes, so our future work will include the study on occlusion invariance.

REFERENCES

[1]. *Xiao X, Wang X B, Wang S J*, "Algorithm based on edge point feature for image matching", Journal of Chinese Computer Systems, **vol. 33**, no.11, 2012, pp. 2535-2537.

[2]. *Wang M, Niu S, Yang X*, "A novel panoramic image stitching algorithm based on ORB", In Proceedings of the International Conference on Applied System Innovation, 2017, pp. 818-821.

[3]. *Furukawa Y, Ponce J*, "Accurate, dense, and robust multiview stereopsis", IEEE Transactions on Pattern Analysis and Machine Intelligence, **vol. 32**, no. 8, 2009, pp. 1362-1376.

[4]. *Lowe D G*, "Distinctive image features from scale-invariant keypoints", International Journal of Computer Vision, **vol. 60**, no.2, 2004, pp. 91-110.

[5]. *Dong J, Soatto S*, "Domain-size pooling in local descriptors: DSP-SIFT", In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 5097-5106.

[6]. *Zagoruyko S, Komodakis N*, "Learning to compare image patches via convolutional neural networks", In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 4353-4361.

[7]. *Tian Y, Fan B, Wu F*, "L2-Net: Deep learning of discriminative patch descriptor in Euclidean space", In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 661-669.

[8]. *Mishchuk A, Mishkin D, Radenovic F, et al.*, "Working hard to know your neighbor's margins: Local descriptor learning loss", In Proceedings of the Neural Information Processing Systems, 2017, pp. 4826-4837.

[9]. *Wang Z, Wu F, Hu Z*, "MSLD: A robust descriptor for line matching", Pattern Recognition, **vol. 42**, no. 5, 2009, pp. 941-953.

[10]. *Zhang L, Koch R*, "An efficient and robust line segment matching approach based on LBD descriptor and pairwise geometric consistency", Journal of Visual Communication and Image Representation, **vol. 24**, no. 7, 2013, pp. 794-805.

[11]. *Liu H, Zhi S, Wang Z*, "IOCD: intensity order curve descriptor", International Journal of Pattern Recognition and Artificial Intelligence, **vol. 27**, no. 7, 2013, pp. 1355011-135037.

[12]. *Pan S J, Yang Q*, "A survey on transfer learning", IEEE Transactions on Knowledge and Data Engineering, **vol. 22**, no. 10, 2010, pp. 1345-1359.

[13]. *Brown M, Hua G, Winder S*, "Discriminative learning of local image descriptors", IEEE Transactions on Pattern Analysis and Machine Intelligence, **vol. 33**, no. 1, 2010, pp. 43-57.

[14]. *Saxe A M, McClelland J L, Ganguli S*, "Exact solutions to the nonlinear dynamics of learning in deep linear neural networks", In Proceedings of the International Conference on Learning Representations, 2013.

[15]. *Xie M, Jean N, Burke M, et al.*, "Transfer learning from deep features for remote sensing and poverty mapping", In Proceedings of the AAAI Conference on Artificial Intelligence, 2016, pp. 3929-3935.

[16]. *Nam H, Han B*, "Learning multi-domain convolutional neural networks for visual tracking", In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 4293-4302.

[17]. *Tajbakhsh N, Shin J Y, Gurudu S R, et al.*, "Convolutional neural networks for medical image analysis: full training or fine tuning?", IEEE Transactions on Medical Imaging, **vol. 35**, no.5 , 2016, pp. 1299-1312.

[18]. PyTorch. http://pytorch.org.

[19]. *Mikolajczyk K, Schmid C*, "A performance evaluation of local descriptors", In Proceedings of the IEEE Transactions on Pattern Analysis and Machine Intelligence, 2003, pp. 257-263.