RF-Net: An End-to-End Image Matching Network based on Receptive Field

Xuelun Shen[1] Cheng Wang[1*] Xin Li[2] Zenglei Yu[1]
Jonathan Li[1,3] Chenglu Wen[1] Ming Cheng[1] Zijian He[1]
[1]Fujian Key Laboratory of Sensing and Computing for Smart City,
School of Information Science and Engineering, Xiamen University, China
[2]School of Electrical Engineering and Computer Science, Louisiana State University, USA
[3]Department of Geography and Environmental Management, University of Waterloo, Canada
{cwang,junli,clwen,chm99}@xmu.edu.cn, xinli@cct.lsu.edu
{xuelun,zengleiyu,kemoho}@stu.xmu.edu.cn,

Abstract

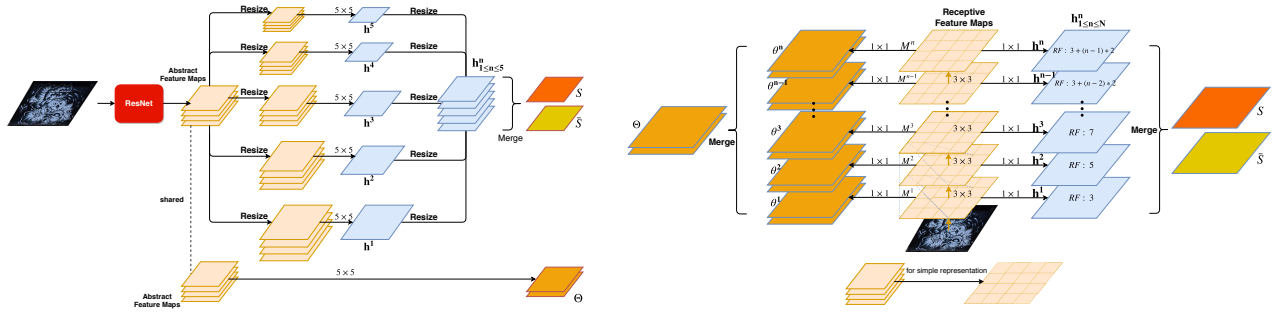
This paper proposes a new end-to-end trainable matching network based on receptive field, RF-Net, to compute sparse correspondence between images. Building end-to-end trainable matching framework is desirable and challenging. The very recent approach, LF-Net, successfully embeds the entire feature extraction pipeline into a jointly trainable pipeline, and produces the state-of-the-art matching results. This paper introduces two modifications to the structure of LF-Net. First, we propose to construct receptive feature maps, which lead to more effective keypoint detection. Second, we introduce a general loss function term, neighbor mask, to facilitate training patch selection. This results in improved stability in descriptor training. We trained RF-Net on the open dataset HPatches, and compared it with other methods on multiple benchmark datasets. Experiments show that RF-Net outperforms existing state-of-the-art methods.

1. Introduction

Establishing correspondences between images plays a key role in many Computer Vision tasks, including but not limited to wide-baseline stereo, image retrieval, and image matching. A typical feature-based matching pipeline consists of two components: detecting keypoints with attributions (scales, orientation), and extracting descriptors. Many existing methods focus on building/training keypoint detectors or feature descriptors individually. However, when integrating these separately optimized subcomponents into a matching pipeline, individual performance gain may not directly add up [29]. Jointly training detectors and descriptors to make them optimally cooperate with each other, hence, is more desirable. However, training such a network is difficult because the two subcomponents have their individually different objectives to optimize. Not many successful end-to-end matching pipelines have been reported in literatures. LIFT [29] is probably the first notable design towards this goal. However, LIFT relies on the output of SIFT detector to initialize the training, and hence, its detector behaves similarly to the SIFT detector. The recent network, SuperPoint [5], achieves this end-to-end training. But its detector needs to be pre-trained on synthetic image sets, and whole network is trained using images under synthesized affine transformations. The more recent LF-Net [18] is inspired by Q-learning, and uses a Siamese architecture to train the entire network without the help of any hand-craft method. In this paper, we develop an end-to-end matching network with enhanced detector and descriptor training modules, which we elaborate as follows.

Keypoint Detection. Constructing response maps is a general way to find keypoints. LIFT [29] obtains response maps by directly applying convolutions on different resolutions of the input image. SuperPoint [5] does not build response maps, but it processes input image using some convolution and max-pooling layers to produce an intermediate tensor $\mathcal{B}$ whose width and height are only $\frac{1}{8}$ of the input. Hence the response on $\mathcal{B}$ represents a highly abstract feature of the input image and the size of the feature's receptive field is larger than 8 pixels. LF-Net uses ResNet [9] to produce abstract feature maps from the input image, then build response maps by convolution on the abstract feature maps at different resolutions. Therefore, the response on each map has a large receptive field. In this work, we build response maps using *concerned receptive fields*. Specifically, we apply convolution to produce feature maps related to the increasing receptive field (Figure 1 (b)). For exam-

*Corresponding author.



(a) LF-Det (b) RF-Det

Figure 1. Scale-space response maps $\mathbf{h}^n_{1\leq n\leq N}$ construction in LF-Det (detector in LF-Net [18]) and RF-Det (detector in our RF-Net). (a) LF-Det constructs response maps using abstract feature maps extracted from ResNet [9]. (b) Our RF-Det constructs response maps using receptive feature maps. Note that RF on $\mathbf{h}^n$ represents the receptive field size.

ple, applying convolution with a kernel size of 3×3 and stride of 1, the receptive field will increase to 3, 5, 7 and so on. This design produces more effective response maps for keypoints detection.

Feature Descriptor. Training *descriptors* in an end-to-end network is very different from training individual ones. Existing (individual) descriptor training is often done on well-prepared datasets such as the *Oxford* Dataset [15], UBC *PhotoTour* Dataset [28], and *HPatches* Dataset [1]. In contrast, in the end-to-end network training, patches need to be produced from scratch. In LF-Net, patch pairs are sampled by rigidly transforming patches surrounding keypoints in image I_i to image I_j. However, a defect of this simple sampling strategy could affect the descriptor training. Specifically, two originally far-away keypoints, after transformed, could become very close to each other. As a result, a negative patch could look very similar to an anchor patch and positive patch. This will confuse the network during training. This situation brings labeling ambiguity and effect descriptor training. We propose a general loss function term called *neighbor mask* to overcome this issue. *Neighbor mask* can be used in both *triplet loss* and its variants.

Integrating our new backbone detector and the descriptor network, our sparse matching pipeline is also trained in an end-to-end manner, without involving any hand-designed component. We observe that the descriptor's performance greatly influences the detector's training, and a more robust descriptor helps detector learn better. Therefore, in each training iteration, we train descriptor twice and detector once. To show the effectiveness of our approach through comprehensive and fair evaluations, we compare our RF-Net with other methods with three evaluation protocols in two public datasets, *HPatches* [1] and *EF* Dataset [34]. Matching experiments demonstrate that our RF-Net outperforms existing state-of-the-art approaches.

The main contributions of this paper are in three aspects. (1) We propose a new receptive field based detector, which generates more effective scale space and response maps. (2) We propose a general loss function term for descriptor learning which improves the robustness of patch sampling. (3) Our integrated RF-Net supports effective end-to-end training, which leads to better matching performance than existing approaches.

2. Related work

A typical feature-based matching pipeline consists of two components: detecting keypoints with attributions (scales, orientation), and extracting descriptors. Many recent learning based pipelines focus on improving one of these modules, such as feature detection [22, 33, 19, 26], orientation estimation [30] and descriptor representation [17, 24, 8]. The deficiency of these approaches is that the performance gain from one improved component may not directly correspond to the improvement of the entire pipeline [29, 23].

Hand-crafted approaches like SIFT [14], is probably the most well-known traditional local feature descriptor. A big limitation of SIFT is the speed. SURF [3] approximates LoG use a box filter and significantly speeds up the detection. Other popular hand-crafted features include WADE [21], Edge Foci [34], Harris corners [7] and its affine-covariant [16].

Many effective **machine-learned detectors** have also been proposed recently. FAST [19] and ORB [20] use machine learning approach to speed up the process of corner detection. TILDE [26] learns from pre-aligned images of the same scene at different illumination conditions. Although being trained with the assistance from SIFT, TILDE can still identify keypoints missed by SIFT, and perform better than SIFT on the evaluated datasets. Quad-Network [22] is trained unsupervisedly with a "ranking" loss. [32] combines this "ranking" loss with a "Peakedness" loss and produces a more repeatable detector. Lenc *et al.* [13] proposes to train a feature detector directly from the covariant constraint. Zhang *et al.* [33] extends the covariant

constraint by defining the concepts of "standard patch" and "canonical feature". The method of [30] learns to estimate orientation to improve feature point matching.

Descriptor learning is the focus of many work for image alignment. DeepDesc [27] applies a Siamese network, MatchNet [6] and Deepcompare [31], to learn nonlinear distance matrix for matching. A series of recent works have considered more advanced model architectures and triplet-based deep metric learning formulations, including UCN [4], TFeat [2], GLoss [12], L2-Net [24], HardNet [17] and He *et al.* [8]. Recent works focus on designing better loss functions, while still using the same network architecture proposed in L2-Net [24].

Building **end-to-end matching frameworks** have been less explored. LIFT [29] was probably the first attempt to build such a network. It combines three CNNs (for the detector, orientation estimator, and descriptor) through differentiable operations. While it aims to extract an SfM-surviving subset of DoG detections, its detector and orientation estimator are fed with a patch instead of the whole image, and hence, are not trained end-to-end. SuperPoint [5] trains a fully-convolutional neural network that consists of a single shared encoder and two separate decoders (for feature detection and description respectively). Synthetic shapes are used to generate images for detector's pre-training, and synthetic homographic transformations are used to produce image pairs for detector's fine-tuning. The more recent LF-Net [18] presents a novel deep architecture and a training strategy to learn a local feature pipeline from scratch. Based on a Siamese Network structure, LF-Net predicts on one branch, and generates ground truth on another branch. It is fed with a QVGA sized image and produces multi-scale response maps. Next, it processes the response maps to output three dense maps, representing keypoints saliency, scale, and orientation, respectively.

3. Approach

Our RF-Net consists of a detector, called RF-Det, which is based on receptive feature maps, and a description extractor whose architecture is the same as L2-Net [24], but with a modified loss function. The design of the whole network structure is depicted in Figure 2. During testing, the detector network RF-Det takes in an image and outputs a score map $\mathbf{S}$, an orientation map $\mathbf{\Theta}$, and a scale map $\bar{\mathbf{S}}$. These three maps produce the locations, orientations, and scales of keypoints, respectively. Patches cropped from these maps will be fed to the descriptor module to extract fixed-length feature vectors for matching.

3.1. Scale Space Response Maps Construction

Constructing scale space response maps is the basis for keypoint detection. We denote the response maps as $\{\mathbf{h}^n\}$, where $1 \leq n \leq N$ and N is the total layer number. The LF-Net [18] uses abstract feature maps extracted from ResNet [9] to construct its response maps. Each response in the abstract feature maps represents a high-level feature extracted from a large region in the image, while the low-level features are not extracted. Thus, every map in $\mathbf{h}^n$ is a large-scale response in the scale space.

Our idea is to preserve both high-level and low-level features when constructing the response maps $\{\mathbf{h}^n\}$, and use some maps (e.g., with smaller index) to offer small-scale response, and some others (e.g., with bigger index) to offer large-scale response.

Following this idea, we use N hierarchical convolutional layers to produce feature maps $\{\mathbf{M}^n\}, 1 \leq n \leq N$ with increasing receptive fields. Therefore, each response in $\mathbf{M}^n$ describes the abstracted features extracted from a certain range of the image, and this range increases as the convolution applies. Then we apply one 1×1 convolution on each $\mathbf{M}^n$ to produce response maps $\mathbf{h}^n_{1 \leq n \leq N}$ in the multi scale space.

In our implementation, we set $N = 10$. And the hierarchical convolutional layers consist of sixteen 3×3 kernels followed by an instance normalization [25] and leaky ReLU activations. We also add shortcut connection [9] between each layer, which does not change the receptive field in feature maps and makes training of the network easier. To produce multi-scale response maps $\mathbf{h}^n$, we use one 1×1 kernel followed by an instance normalization. All convolution are zero-padded to make the output size same as the input.

3.2. Keypoint Detection

Following the commonly adopted strategy, we select high-response pixels as keypoints. Response maps $\mathbf{h}^n$ represent pixels' response on multi-scales, so we produce the keypoint score map from it. Then we design the keypoint detection similar to LF-Net [18], except that our response maps $\mathbf{h}^n$ are constructed by receptive feature maps.

Specifically, we perform two softmax operators to produce the score map $\mathbf{S}$. The purpose of the first softmax operator is to produce sharper response maps $\hat{\mathbf{h}}^n$. The first softmax operator is applied over a $15 \times 15 \times N$ window sliding on $\mathbf{h}^n$ with the same zero padding. Then we merge all the $\hat{\mathbf{h}}^n$ into the final score map $\mathbf{S}$ with the second $softmax_n$ operator, by

$$Pr^n = softmax_n(\hat{\mathbf{h}}^n), \tag{1}$$

and

$$\mathbf{S} = \sum_n \hat{\mathbf{h}}^n \odot Pr^n, \tag{2}$$

where $\odot$ is the Hadamard product, and Pr^n indicates the probability of a pixel being a keypoint. The second $softmax_n$ is applied on a $1 \times 1 \times N$ window sliding on $\hat{\mathbf{h}}^n$.

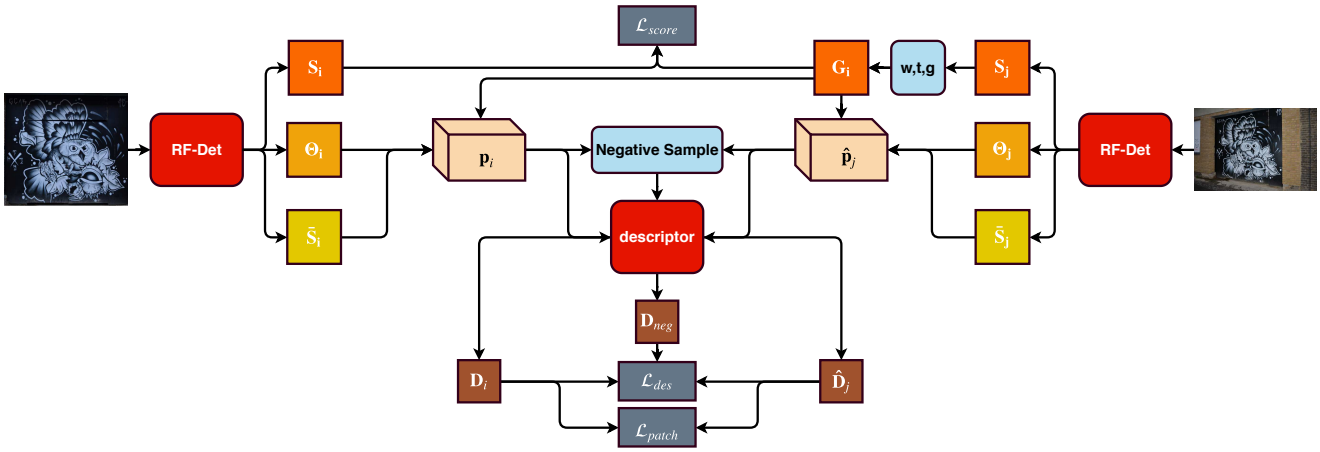


Figure 2. The whole network structure for RF-Net. In training, we feed one pair of images into the network. The image on the right is processed by the network to generate the ground truth of the left image. After calculating the gradient of the loss function, parameters are updated by back propagation. Next we exchange the positions of the two images and train the network again.

Estimations of the orientation and scale are also produced based on Pr^n. We apply convolutions on $\mathbf{M}^n$ with two 1×1 kernels to produce multi-scale orientation maps $\{\theta^n\}$ (see Figure 1 (b)) whose values indicate the $sine$ and $cosine$ of the orientation. The values are used to compute an angle using the $arctan$ function. Then we apply the same product to merge all θ^n into the final orientation map $\mathbf{\Theta}$, by

$$\mathbf{\Theta} = \sum_n \theta^n \odot Pr^n. \tag{3}$$

To produce the scale map $\bar{\mathbf{S}}$, we apply the similar operation used in orientation estimation:

$$\bar{\mathbf{S}} = \sum_n \bar{\mathbf{s}}^n \odot Pr^n, \tag{4}$$

where $\bar{\mathbf{s}}^n$ is the receptive field size of the $\mathbf{h}^n$.

3.3. Descriptor extraction

We develop the descriptor extraction module in the network following a structure similar to the L2-Net [24]. This structure is also adopted in other recent descriptor learning frameworks such as Hard-Net [17] and He *et al.* [8]. Specifically, this descriptor network consists of seven convolution layers, each followed by a batch normalization and ReLU, except for the last one. The output descriptors are L2 normalized, and its dimension is 128. We denote the output descriptors as $\mathbf{D}$. While we adopt this effective network structure similar to many recent descriptor extraction modules, we use a different loss function, which is discussed in the following.

3.4. Loss Function

A keypoint detector predicts keypoints' locations, orientations, and scales. Therefore, its loss function consists of *score loss* and *patch loss*. Patch descriptor is independent from the detection component, once the keypoints are selected. Hence, we use another *description loss* to train it.

Score loss. In this feature matching problem, because it is unclear which points are important, we cannot produce ground truth score maps through human labeling. Good detectors should be able to find the corresponding interest points when the image undergoes a transformation. A simple approach is to let the two score maps $\mathbf{S}_i$ and $\mathbf{S}_j$ (produced from images I_i and I_j, respectively) to have the same score at the corresponding locations. A simple approach to implement the idea is to minimize Mean Square Loss (MSE) between corresponding locations on $\mathbf{S}_i$ and $\mathbf{S}_j$. However, this approach turned out to be not very effective in our experiments.

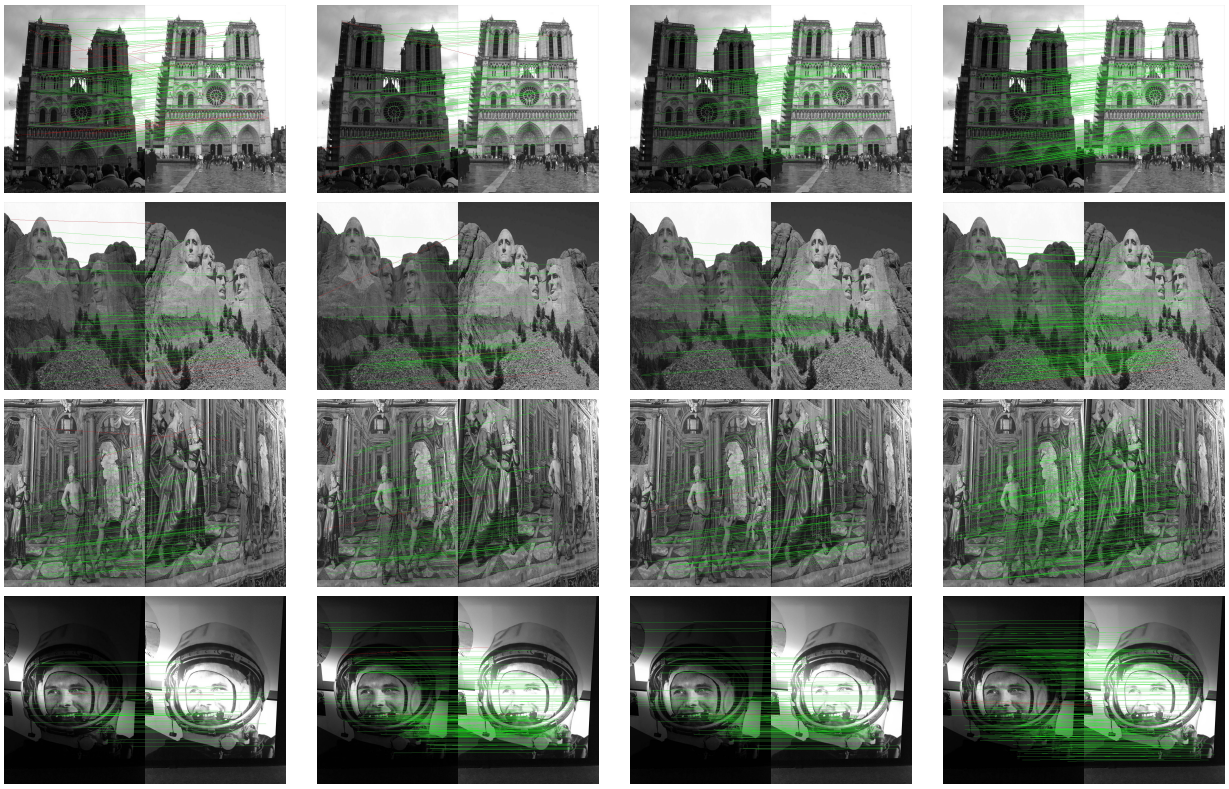
LF-Net suggests another approach. We fed image pair I_i and I_j into network to produce $\mathbf{S}_i$ and $\mathbf{S}_j$. We process $\mathbf{S}_j$ to produce ground truth $\mathbf{G}_i$, then define the score loss to be the MSE between $\mathbf{S}_i$ and $\mathbf{G}_i$. More specifically, given the ground truth perspective matrix, first, we select the top K keypoints from the warped score map $\mathbf{S}_j$, and we denote this as operation t. Then, we generate a clean ground truth score map $\mathbf{G}_i$ by placing Gaussian kernels with standard deviation $\sigma = 0.5$ at those locations. This operation is denoted as g. Then for warping, we apply a perspective transform w. This score loss is finally written as:

$$\mathbf{G}_i = g(t(w(\mathbf{S}_j))), \tag{5}$$

$$\mathcal{L}_{score}(\mathbf{S}_i, \mathbf{S}_j) = |\mathbf{S}_i - \mathbf{G}_i|^2. \tag{6}$$

If a keypoint falls outside the image I_i, we drop it from the optimization process.

Patch loss. Keypoint orientation and scale affect the patches cropped from the image; and descriptors extracted



(a) SIFT (b) FAST+Hard-Net (c) LF-net (d) RF-Net

Figure 3. Qualitative matching results, with correct matches drawn in green lines and failed matches drawn in red lines. These columns are SIFT, FAST detector integrated with Hard-Net descriptor, LF-Net, and RF-Net. The images in top two rows are from *EF* Dataset [34], and the images in bottom two rows are from *HPatches* [1]. We use the nearest neighbor distance ratio(0.7) as matching strategy with $K = 1024$ keypoints to match two images. As the figure showed, more green lines and fewer red lines means better matching results.

from patches influence matching precision. We define a *patch loss* to optimize detector to detect more consistent keypoints. We hope that the patches cropped from the corresponding position are as similar as possible.

Specifically, we select the top K keypoints from $\mathbf{G}_i$, then warp their spatial coordinates back to I_j, and form the keypoint with orientation and scale from $\mathbf{\Theta}$ and $\bar{\mathbf{S}}$ predicted by each image. We extract descriptors $\mathbf{D}_i^k$ and $\hat{\mathbf{D}}_j^k$ at these corresponding patches $\mathbf{p}_i^k$ and $\hat{\mathbf{p}}_j^k$. The patch loss can be formulated as

$$\mathcal{L}_{patch}(\mathbf{D}_i^k, \hat{\mathbf{D}}_j^k) = \frac{1}{K} \sum_{k=1,K} d(\mathbf{D}_i^k, \hat{\mathbf{D}}_j^k), \tag{7}$$

where

$$d(x, y) = \sqrt{2 - 2xy}. \tag{8}$$

Unlike LF-Net that selects keypoints from $\mathbf{I}_i$, we select keypoints from $\mathbf{G}_i$. This is because in many public training datasets (e.g. HPatches), there is no background mask available. After transformed, Keypoints selected from $\mathbf{I}_i$ may be out of range on image $\mathbf{I}_j$. Therefore, the training data sampling method we use is more general.

Description loss. The description loss we use is based on the *hard loss* proposed in Hard-Net [17]. The *hard loss* maximizes the distance between the closest positive and closest negative example in the batch. Considering the patches sampled from scratch may bring label ambiguity, we improve the *hard loss* by a *neighbor mask*, which makes descriptor training more stable. We formulate description loss as

$$\mathcal{L}_{des}(\mathbb{D}_{pos}, \mathbb{D}_{ng}) = \frac{1}{K} \sum max(0, 1 + \mathbb{D}_{pos} - \mathbb{D}_{ng}), \tag{9}$$

where

$$\mathbb{D}_{pos}(\mathbf{D}_i^k, \hat{\mathbf{D}}_j^k) = d(\mathbf{D}_i^k, \hat{\mathbf{D}}_j^k), \tag{10}$$

and

$$\mathbb{D}_{ng} = min(d(\mathbf{D}_i^k, \hat{\mathbf{D}}_j^n), d(\mathbf{D}_i^m, \hat{\mathbf{D}}_j^k)). \tag{11}$$

Here $\hat{\mathbf{D}}_j^n$ is the closest non-matching descriptor to $\mathbf{D}_i^k$ where

$$n = argmin_{n' \neq k} d(\mathbf{D}_i^k, \hat{\mathbf{D}}_j^{n'}) \;\&\; E(\hat{\mathbf{p}}_j^k, \mathbf{p}_j^{n'}) > \mathbf{C}. \tag{12}$$

$\mathbf{D}_i^m$ is the closest non-matching descriptor to $\hat{\mathbf{D}}_j^k$ where

$$m = argmin_{m' \neq k} d(\mathbf{D}_i^{m'}, \hat{\mathbf{D}}_j^k) \& E(\mathbf{p}_i^{m'}, \mathbf{p}_i^k) > \mathbf{C}. \tag{13}$$

Function E computes the Euclidean distance between the centroids of the two patches. We call it a *neighbor mask*. If a patch $\mathbf{p}_i^{m'}$ is very close to $\mathbf{p}_i^k$, then $\mathbf{p}_i^{m'}$ and $\hat{\mathbf{p}}_j^k$ should be a correct match. If a patch $\hat{\mathbf{p}}_j^{n'}$ is very close to $\hat{\mathbf{p}}_j^k$, then $\hat{\mathbf{p}}_j^{n'}$ and $\mathbf{p}_i^k$ should be a correct match. Therefore, we call patch $\mathbf{p}_i^{m'}$ a positive patches of $\hat{\mathbf{p}}_j^k$ if their centroid distance is less than a threshold $\mathbf{C}$. We mask it when collecting negative samples for $\hat{\mathbf{p}}_j^k$.

In summary, we train description network with $\mathcal{L}_{des}$ and train detection network with $\mathcal{L}_{det}$:

$$\mathcal{L}_{det} = \lambda_1 \mathcal{L}_{score} + \lambda_2 \mathcal{L}_{patch}. \tag{14}$$

4. Experiments

4.1. Training

Training data. We trained our network on open dataset *HPatches* [1]. This is a recent dataset for local patch descriptor evaluation consists of 116 sequences of 6 images with known homography. The dataset is split into two parts: $viewpoint$ - 59 sequences with significant viewpoint change and $illumination$ - 57 sequences with significant illumination change, both natural and artificial. We split the viewpoint sequences by a ratio of 0.9 (53 sequences for training and validation, and rest 6 sequences for testing).

At training stage, we resized all images into 320×240, then converted images to gray for simplicity and normalized them individually using their mean and standard deviation. Differ with LF-Net [18], we do not have depth maps for each image, so all pixels in the image were used for training.

About the training patches for description extractor, we cropped image patches and resized them to 32×32 by selecting the top K keypoints with their orientation and scale. To keep differentiability, we used a bilinear sampling scheme of [10] for cropping.

Training detail. At training stage, we extracted $K = 512$ keypoints for training, but at the testing stage, we can choose as many keypoints as desired. For optimization, we used ADAM [11], and set initial learning rate 0.1 both for detector and descriptor, and trained descriptor twice and then trained detector once. The $\mathbf{C}$ in *neighbor mask* is 5.

4.2. Evaluation data and protocol

Beside *HPatches* illumination and viewpoint sequences, we also evaluated our model on *EF* Dataset [34]. *EF* Dataset has 5 sequences of 38 images which contains drastic illumination and background clutter changes.

The definition of a match depends on the matching strategy. To evaluate the entire local feature pipeline performance, we use three matching strategies from [15] to calculate match score for quantitative evaluation:

- The first is nearest neighbor (NN) based matching, two regions $\mathbf{A}$ and $\mathbf{B}$ are matched if the descriptor $\mathbf{D_B}$ is the nearest neighbor to $\mathbf{D_A}$. With this approach, a descriptor has only one match.
- The second is nearest neighbor with a threshold (NNT) based matching, two regions $\mathbf{A}$ and $\mathbf{B}$ are matched if the descriptor $\mathbf{D_B}$ is the nearest neighbor to $\mathbf{D_A}$ and if the distance between them is below a threshold t.
- The third is nearest neighbor distance ratio (NNR) based matching, two regions $\mathbf{A}$ and $\mathbf{B}$ are matched if $||\mathbf{D_A} - \mathbf{D_B}|| \ / \ ||\mathbf{D_A} - \mathbf{D_C}|| < t$, where $\mathbf{D_B}$ is the first and $\mathbf{D_C}$ is the second nearest neighbor to $\mathbf{D_A}$.

	HP-illum	HP-view	EF
SIFT	0.490	0.494	0.296
SURF	0.493	0.481	0.235
L2-Net+DoG	0.403	0.394	0.189
L2-Net+SURF	0.627	0.629	0.307
L2-Net+FAST	0.571	0.431	0.229
L2-Net+ORB	0.705	0.673	0.298
L2-Net+Zhang et al.	0.685	0.425	0.235
Hard-Net+DoG	0.436	0.468	0.206
Hard-Net+SURF	0.650	0.668	0.334
Hard-Net+FAST	0.617	0.630	0.290
Hard-Net+ORB	0.616	0.632	0.238
Hard-Net+Zhang et al.	0.671	0.557	0.273
LF-Net	0.617	0.566	0.251
RF-Net	**0.783**	**0.808**	**0.453**

Table 1. **Comparison to state-of-the-art and baselines.** Average match score measured with three evaluation protocol in each image sequences. Individual descriptors and detector are trained under same sequences as end-to-end networks. All feature descriptors is 128 dimension and L2-Normalized.

All matching strategies compared each descriptor of the reference image with each descriptor of the transformed image. To emphasize accurate localization of keypoints, follow [18, 19], we used 5-pixel threshold instead of the overlap measure used in [15]. All learned descriptors have been L2 normalized and their distance range is at $[0, 2]$. For fairness, we also L2 normalized hand-craft descriptors and set 1.0 as the nearest neighbor threshold and 0.7 as the nearest neighbor distance ratio threshold.

4.3. Results on match performance

We compared RF-Net to three types of methods, the first one is full local feature pipelines, SIFT [14], SURF [3], LF-Net [18]. The second one is hand-craft detector integrated with learned descriptor, that is DoG [14], SURF [3] FAST [19] and ORB [20] integrated with L2-Net [24] and Hard-Net [17]. The third one is learned detector integrated with a learned descriptor, that is Zhang *et al.* [33] integrated with L2-Net [24] and Hard-Net [17]. We use the authors' release for L2-Net, Hard-Net, LF-Net and Zhang *et*

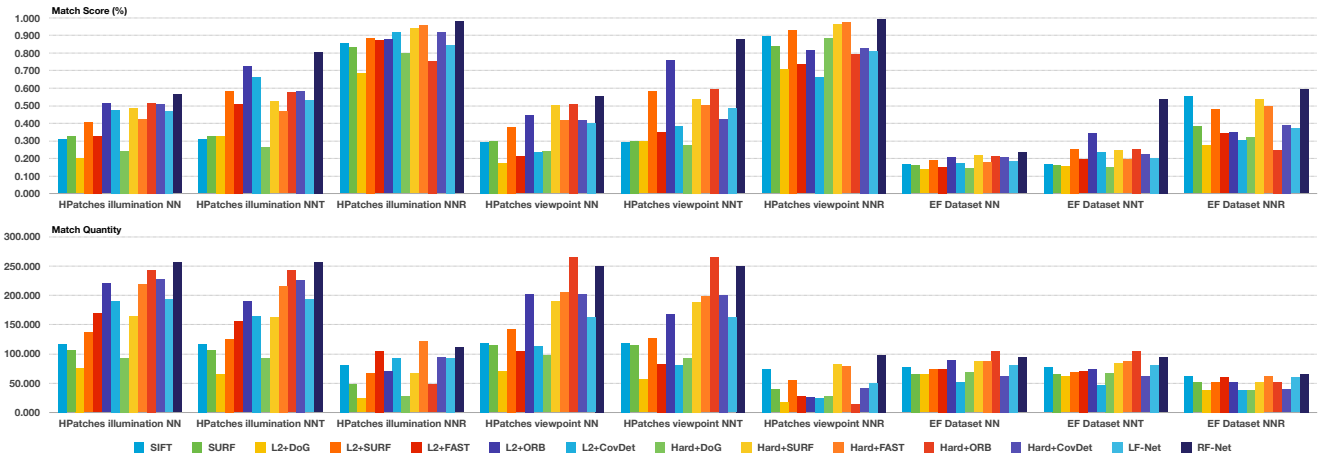


Figure 4. **Top**: average match score in each evaluation protocol and sequences. **Bottom**: average match quantity in each evaluation protocol and sequences. The match score of RF-Net outperforms the closest competitor by a large margin in *EF* Dataset NNT.

	HP-illum	HP-view	EF
LF-Det+LF-Des	0.617	0.566	0.251
RF-Det+LF-Des	0.720	0.665	0.325
LF-Det+RF-Des	0.744	0.714	0.361
RF-Det+RF-Des	**0.783**	**0.808**	**0.453**

Table 2. **Ablation studies.** Average match score measured with three evaluation protocol in each image sequences. All methods are trained end-to-end with the same training data. LF-Des represents the descriptor used in LF-Net [18], and RF-Des represents the descriptor used in our RF-Net. The pipeline performance improved by replacing LF-Det with RF-Det.

	HP-illum	HP-view	EF
RF-Net(No Mask)	0.734	0.753	0.423
RF-Net(No Orient)	0.762	0.791	0.432
RF-Net	**0.783**	**0.808**	**0.453**

Table 3. **Ablation studies.** Average match score measured with three evaluation protocol in each image sequences. RF-Net(No Mask) means RF-Net trained without the *neighbor mask* loss function item. RF-Net(No Orient) means RF-Net trained without orientation estimation module.

al., and OpenCV for the rest. For LF-Net and Zhang *et al.*, we trained them same as RF-Net in 53 viewpoint image sequences cut from *HPatches* [1]. For Hard-Net and L2-Net, we trained them in 53 viewpoint patches sequences provided by *HPatches*. The length of all feature descriptors is 128 dimension and L2-Normalized.

As shown in Table. 1, our RF-Net outperforms all others and sets the new state-of-the-art on *HPatches* and *EF* Dataset. Our RF-Net outperforms the closest competitor by 11%, 20% and 35% relative in the three sequences.

Match score represents the correct ratio in method prediction, while match quantity represents the correct predicted quantity. Figure. 4 depicts the match score and match quantity in all evaluations, and our RF-Net get both high match score and match quantity. The pipeline of ORB combined with Hard-Net also achieves good match quantity in NN and NNT protocols, but it does not perform well in NNR protocol. This indicates descriptors extracted by this pipeline have high nearest neighbor distance ratio, while our RF-Net does not have this problem.

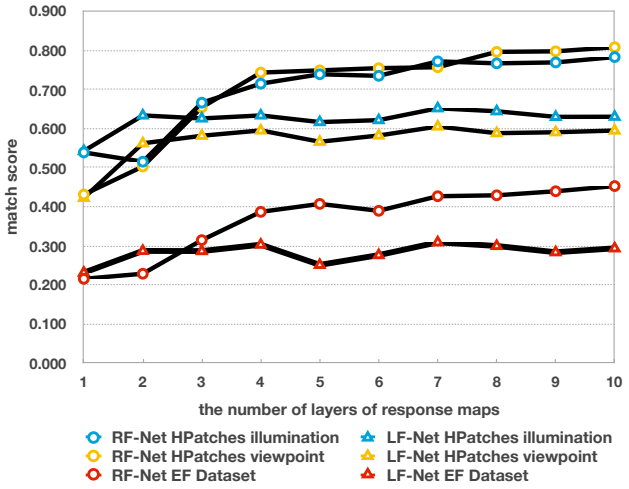
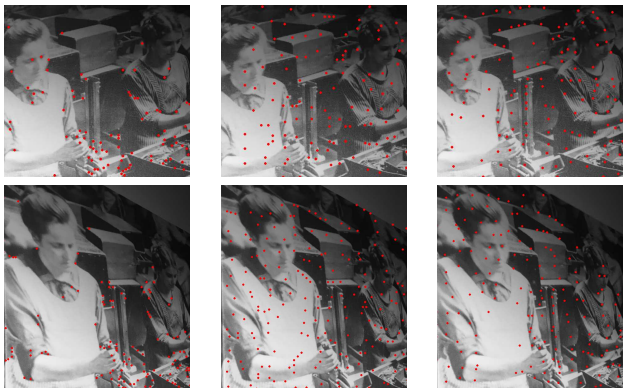


Figure 5. Match score comparison of RF-Net and LF-Net under different N-layer response maps.

We also give the experiment results about how N response layers effect the RF-Net and LF-Net in Figure. 5. For RF-Net, match score increases with the number of response layers and saturates after $N = 8$, and the gap in performance between the LF-Net and RF-Net starts from $N = 3$ and increases as N increases. This demonstrates that receptive field based response maps are more effective than abstract feature based method

	HPatches-illum			HPatches-view			EF Dataset			
#keypoints	512	1024	2048	512	1024	2048	512	1024	2048	Average
DoG	0.638	0.672	0.687	0.609	0.645	0.646	0.512	0.572	0.572	0.617
FAST	0.790	0.853	0.899	0.704	**0.801**	**0.853**	0.560	**0.691**	**0.791**	**0.771**
ORB	0.780	0.830	0.869	**0.709**	0.769	0.821	0.524	0.598	0.669	0.730
SURF	0.708	0.746	0.746	0.633	0.665	0.665	0.569	0.613	0.613	0.662
Zhang et al.	**0.827**	**0.894**	0.917	0.516	0.664	0.747	**0.588**	0.638	0.638	0.714
LF-Det	0.727	0.854	**0.922**	0.558	0.650	0.717	0.507	0.586	0.667	0.688
RF-Det	0.793	0.868	0.889	0.689	0.723	0.729	0.575	0.669	0.704	0.738

Table 4. Repeatability at different keypoints in three evaluation sequences.



(a) FAST (b) LF-Det (c) RF-Det

Figure 6. Visualize keypoints detected by FAST, LF-Det and RF-Det. Keypoints detected by RF-Det and LF-Det are more sparse than FAST.

4.4. Discussions and ablation studies

In this section, we examine the importance of various components of our architecture. We replaced LF-Det with RF-Det, and trained them with the same training data to show the effectiveness of our RF-Det. Table. 2 shows the pipeline performance improved by replacing LF-Det with RF-Det.

To mine the effectiveness of modules in RF-Net, we try to remove *neighbor mask* and orientation estimation module from RF-Net. Table. 3 shows *neighbor mask* brings remarkable match improvement to RF-Net. Even we removed orientation prediction, our RF-Net still gets state-of-the-art match score, this represents the robustness of our RF-Det.

4.5. Results on repeatability

Table. 4 shows the repeatability performance of handcraft approaches, Zhang *et al.*, LF-Det and our RF-Det. Although FAST does not perform best on image match, it gets the highest repeatability. Pipeline of matching is a cooperation task between detector and descriptor. As shown in Figure. 6, the keypoints detected by learned end-to-end detector (LF-Det and RF-Det) are more sparse than FAST. This indicates sparse keypoints are easier to match, because too close keypoints may produce patches too similar to match. Therefore, a parse detector works better on this task. Compare RF-Det with LF-Det, RF-Det indeed gets a higher repeatability than LF-Det in all sequences. This also benefited from the receptive field design.

4.6. Qualitative results

In Figure. 3, We also give some qualitative results on the task of matching challenging pairs of images provided by *EF* Dataset and *HPatches*. We selected top $K = 1024$ keypoints firstly, then matched them by the nearest neighbor distance ratio matching strategy with 0.7 threshold. We compared our method with the SIFT [14], FAST [19] detector integrated with Hard-Net [17], and LF-Net [18]. The images in top two rows are from *EF* Dataset, and the images in bottom two rows are from *HPatches*. These images are under large illumination changes or perspective transformation. As shown in Figure. 3, our method produced the maximum quantity of green matching lines and fewer red failed match lines.

5. Conclusions

We present a novel end-to-end deep network, RF-Net, for local feature detection and description. To learn more robust response maps, we propose a novel keypoint detector based on receptive field. We also design a loss function term, *neighbor mask*, to learn a more stable descriptor. Both of these designs bring significant performance improvement to the matching pipeline. We conducted qualitative and quantitative evaluations in three data sequences and showed significant improvements over existing state-of-the-art.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. U1605254, 61728206) and the National Science Foundation of USA EAR-1760582.

References

[1] Vassileios Balntas, Karel Lenc, Andrea Vedaldi, and Krystian Mikolajczyk. Hpatches: A benchmark and evaluation of handcrafted and learned local descriptors. *CVPR*, pages 3852–3861, 2017.

[2] Vassileios Balntas, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. Learning local feature descriptors with triplets and shallow convolutional neural networks. In *BMVC*, 2016.

[3] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer Vision and Image Understanding*, 110:346–359, 2008.

[4] Christopher Bongsoo Choy, JunYoung Gwak, Silvio Savarese, and Manmohan Krishna Chandraker. Universal correspondence network. In *NIPS*, 2016.

[5] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. *arXiv:1712.07629*, 2017.

[6] Xufeng Han, Thomas Leung, Yangqing Jia, Rahul Sukthankar, and Alexander C. Berg. Matchnet: Unifying feature and metric learning for patch-based matching. *CVPR*, pages 3279–3286, 2015.

[7] Christopher G. Harris and Mike Stephens. A combined corner and edge detector. In *Alvey Vision Conference*, 1988.

[8] Kun He, Yan Lu, and Stan Sclaroff. Local descriptors optimized for average precision. *CVPR*, pages 596–605, 2018.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CVPR*, pages 770–778, 2016.

[10] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *NIPS*, pages 2017–2025, 2015.

[11] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[12] B. G. Vijay Kumar, Gustavo Carneiro, and Ian D. Reid. Learning local image descriptors with deep siamese and triplet convolutional networks by minimizing global loss functions. *CVPR*, pages 5385–5394, 2016.

[13] Karel Lenc and Andrea Vedaldi. Learning covariant feature detectors. In *ECCV*, pages 100–117. Springer, 2016.

[14] David G Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[15] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *TPAMI*, 27:1615–1630, 2003.

[16] Krystian Mikolajczyk and Cordelia Schmid. Scale & affine invariant interest point detectors. *IJCV*, 60:63–86, 2004.

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

[18] Yuki Ono, Eduard Trulls, Pascal Fua, and Kwang Moo Yi. Lf-net: Learning local features from images. *arXiv preprint arXiv:1805.09662*, 2018.

[19] Edward Rosten, Reid Porter, and Tom Drummond. Faster and better: A machine learning approach to corner detection. *TPAMI*, 32:105–119, 2010.

[20] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary R. Bradski. Orb: An efficient alternative to sift or surf. *ICCV*, pages 2564–2571, 2011.

[21] Samuele Salti, Alessandro Lanza, and Luigi di Stefano. Keypoints from symmetries by wave propagation. *CVPR*, pages 2898–2905, 2013.

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

[23] Johannes L. Schönberger, Hans Hardmeier, Torsten Sattler, and Marc Pollefeys. Comparative evaluation of hand-crafted and learned local features. *CVPR*, pages 6959–6968, 2017.

[24] Yurun Tian, Bin Fan, and Fuchao Wu. L2-net: Deep learning of discriminative patch descriptor in euclidean space. *CVPR*, pages 6128–6136, 2017.

[25] Dmitry Ulyanov, Andrea Vedaldi, and Victor S. Lempitsky. Instance normalization: The missing ingredient for fast stylization. *CoRR*, abs/1607.08022, 2016.

[26] Yannick Verdie, Kwang Moo Yi, Pascal Fua, and Vincent Lepetit. Tilde: A temporally invariant learned detector. *CVPR*, pages 5279–5288, 2015.

[27] Limin Wang, Yu Qiao, and Xiaoou Tang. Action recognition with trajectory-pooled deep-convolutional descriptors. *CVPR*, pages 4305–4314, 2015.

[28] Simon A. J. Winder and Matthew A. Brown. Learning local image descriptors. *CVPR*, pages 1–8, 2007.

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

[30] Kwang Moo Yi, Yannick Verdie, Pascal Fua, and Vincent Lepetit. Learning to assign orientations to feature points. *CVPR*, pages 107–116, 2016.

[31] Sergey Zagoruyko and Nikos Komodakis. Learning to compare image patches via convolutional neural networks. *CVPR*, pages 4353–4361, 2015.

[32] Linguang Zhang and Szymon Rusinkiewicz. Learning to detect features in texture images. In *CVPR*, pages 6325–6333, 2018.

[33] Xu Zhang, Felix X. Yu, Svebor Karaman, and Shih-Fu Chang. Learning discriminative and transformation covariant local feature detectors. *CVPR*, pages 4923–4931, 2017.

[34] C. Lawrence Zitnick and Krishnan Ramnath. Edge foci interest points. *ICCV*, pages 359–366, 2011.