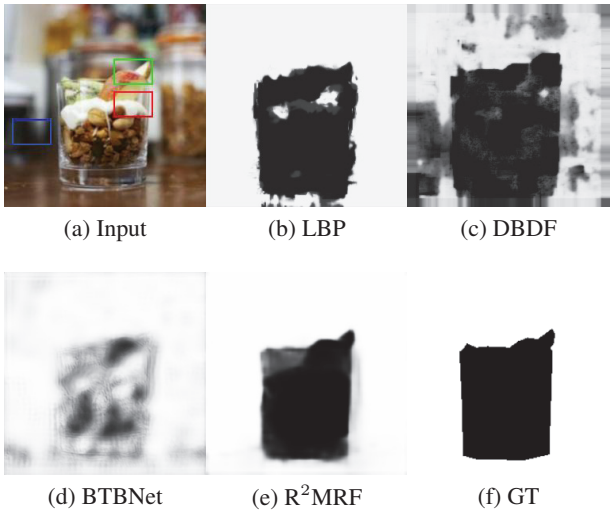
R^{2}MRF: Defocus Blur Detection via Recurrently Refining Multi-Scale Residual Features

Chang Tang,[1] **Xinwang Liu,**[2] **Xinzhong Zhu,**[3] **En Zhu,**[2]
Kun Sun,[1] **Pichao Wang,**[4] **Lizhe Wang,**[1] **Albert Zomaya**[5]
[1]School of Computer Science, China University of Geosciences, Wuhan 430074, China
[2]School of Computer Science, National University of Defense Technology, Changsha 410073, China
[3]College of Mathematics, Physics and Information Engineering, Zhejiang Normal University, Jinhua 321004, China
[4]Alibaba Group (U.S.) Inc, Bellevue, WA 98004, USA
[5]School of Information Technologies, University of Sydney, NSW 2006, Australia
https://github.com/ChangTang/R2MRF

Abstract

Defocus blur detection aims to separate the in-focus and out-of-focus regions in an image. Although attracting more and more attention due to its remarkable potential applications, there are still several challenges for accurate defocus blur detection, such as the interference of background clutter, sensitivity to scales and missing boundary details of defocus blur regions. In order to address these issues, we propose a deep neural network which **R**ecurrently **R**efines **M**ulti-scale **R**esidual **F**eatures (R^{2}MRF) for defocus blur detection. We firstly extract multi-scale deep features by utilizing a fully convolutional network. For each layer, we design a novel recurrent residual refinement branch embedded with multiple residual refinement modules (RRMs) to more accurately detect blur regions from the input image. Considering that the features from bottom layers are able to capture rich low-level features for details preservation while the features from top layers are capable of characterizing the semantic information for locating blur regions, we aggregate the deep features from different layers to learn the residual between the intermediate prediction and the ground truth for each recurrent step in each residual refinement branch. Since the defocus degree is sensitive to image scales, we finally fuse the side output of each branch to obtain the final blur detection map. We evaluate the proposed network on two commonly used defocus blur detection benchmark datasets by comparing it with other 11 state-of-the-art methods. Extensive experimental results with ablation studies demonstrate that R^{2}MRF consistently and significantly outperforms the competitors in terms of both efficiency and accuracy.

Defocus blur is a common phenomenon which occurs when the objects of a scene are not exactly at the camera's focus distance. As an important task, defocus blur detection aims to separate the out-of-focus regions from an image, which has obtained more and more attention due to its wide potential applications such as image deblurring (Shi, Xu, and Jia 2014), defocus magnification (Bae and Durand 2007), image refocusing (Zhang and Cham 2009; 2012), image quality assessment (Wang et al. 2008), and salient object detection (Jiang et al. 2013), just list a few.



(a) Input (b) LBP (c) DBDF

(d) BTBNet (e) R^{2}MRF (f) GT

Figure 1: Some challenging cases for defocus blur detection. (a) Input image, defocus blur detection maps obtained by (b) LBP, (c) DBDF, (d) BTBNet, (e) our R^{2}MRF, and (f) ground truth (GT).

During the past decades, a large number of defocus blur detection methods have been proposed. Based on the used image features, these methods can be generally classified into two categories, i.e., methods based on traditional hand-crafted features and deep learning features. As to the former kind of methods, they often extract low-level features such as gradient and frequency which can model the edge changes since defocus blur usually blunts object edges in an image(Liu, Li, and Jia 2008; Su, Lu, and Tan 2011; Zhuo and Sim 2011; Vu, Phan, and Chandler 2012; Zhang and Hirakawa 2013; Zhu et al. 2013; Shi, Xu, and Jia 2014; Pang et al. 2015; Tang et al. 2016; Saad and Hirakawa 2016; Park et al. 2017). Despite great improvement has been achieved by traditional methods based on hand-crafted features, there are still several challenges which hamper the final results. Firstly, traditional low-level features cannot distinguish the blurry smooth regions which do not contain

structural information from the in-focus smooth regions. Secondly, these methods cannot well capture the global semantic information which is critical for detecting low-contrast focal regions (as shown in the red rectangular region of Figure 1a) and suppressing the background clutter (as shown in the blue rectangular region of Figure 1a). In addition, the edge information of in-focus objects have not been well preserved (as shown in the green rectangular region of Figure 1a).

In the past few years, due to the powerful feature extraction and learning capability, deep convolutional neural networks (CNNs) have been widely used in various computer vision tasks and made significant advances.To this end, CNNs are also leveraged for image defocus blur region detection (Yan and Shao 2016; Park et al. 2017; Zhao et al. 2018). Park et al. (Park et al. 2017) extracted both deep and hand-crafted features in image patches which contain sparse strong edges. However, low-contrast focal regions are still not well distinguished. Recently, Zhao et al. (Zhao et al. 2018) proposed a multi-stream bottom-top-bottom fully convolutional network (BTBNet), which is the first attempt to develop an end-to-end deep network for defocus blur detection. In BTBNet, low-level cues and high-level semantic information are integrated to promote the final results and a multi-stream strategy is leveraged to handle the defocus degree's sensitivity to image scales. Although significant improvement has been obtained by BTBNet, it uses a forward stream and a backward stream to integrate features from different levels for each image scale, which would cause high computational complexity for both network training and testing. More importantly, the complementary information of different layers in this method cannot been fully exploited, which causes some background clutters in the final results. In addition, some low-contrast focal areas are still mistakenly detected as defocus blur regions.

In this work, we propose a novel efficient pixel-wise fully convolutional network for defocus blur detection via recurrently refining multi-layer residual features (R^2MRF). The advantages of the residual learning and multi-level information encoded in multiple layers of a convolutional network are efficiently leveraged to boost the detection results. Specifically, we firstly extract multi-scale deep features by utilizing a backbone fully convolutional network. For each layer, we design a novel recurrent residual refinement branch and embed multiple residual refinement modules (RRMs) into the branch to more accurately detect blur regions from the input image. Considering that the features from bottom layers are able to capture rich low-level features for details preservation while the features from top layers are capable of exploiting the semantic information for locating blur regions, we aggregate the deep features from multiple layers to learn the residual between the intermediate prediction and the ground truth for each recurrent step in each residual refinement branch. Since the image scale significantly influences the clarity of an image, which makes the defocus degree sensitive to image scales, we add the learned residual output of each RRM from each layer to the input of RRM of its previous layer in a hierarchical manner at each recurrent step, and finally fuse the side output of each branch to obtain the final blur detection map. We summarize the technical contributions of this work as follows:

- A new effective and efficient pixel-wise fully convolutional network is proposed to detect defocus regions from a still input image via recurrently refining multi-scale residual features (R^2MRF), which can accurately separate defocus regions from homogeneous and low-contrast focal regions.
- A novel residual refinement module (RRM) is designed to learn the residual between the intermediate prediction and the ground truth at each recurrent step.
- We evaluate the proposed R^2MRF on two commonly used benchmark datasets and compare it with 11 state-of-the-art defocus blur detection methods. Both qualitative and quantitative experimental results validate the superority of our method over other competitors on the two datasets. In addition, our R^2MRF is very efficient and it takes only less than 0.1s by using a single GTX 1080Ti GPU with 32G memory to generate the defocus blur map for a testing image in the two datasets.

Related Work

As a subfield of computer vision, defocus blur detection has been widely investigated due to its important role in many practical applications. Therefore, various defocus blur detection methods have been proposed, which can be generally categorized into two classes, i.e., hand-crafted features based methods and deep learning based methods. Following we give a brief review about these methods.

Hand-crafted Features based Methods

Since defocus blur usually degenerates object edges in an image, traditional methods often extract features such as gradient and frequency which can describe the change of edges (Elder and Zucker 1998; Tai and Brown 2009; Couzinie-Devy et al. 2013; Tang, Hou, and Song 2013). Based on the observation that the first few most significant eigen-images of a blurred image patch usually have higher weights (i.e. singular values) than an image patch with no blur, Su et al. (Su, Lu, and Tan 2011) detected blur regions by examining singular value information for each image pixels. Shi et al. (Shi, Xu, and Jia 2014) studied a series of blur feature representations such as gradient, Fourier domain, and data-driven local filters features to enhance discriminative power for differentiating blurred and unblurred image regions. In (Pang et al. 2015), Pang et al. developed a kernel-specific feature for blur detection, the blur regions and in-focus regions are classified using SVM. Considering that feature descriptors based on local information cannot distinguish the just noticeable blur reliably from unblurred structures, Shi et al. (Shi, Xu, and Jia 2015) proposed a simple yet effective blur feature via sparse representation and image decomposition. Yi and Eramian (Yi and Eramian 2016) designed a sharpness metric based on local binary patterns and the in- and out-of-focus image regions are separated by using the metric. Tang et al. (Tang et al. 2016) designed a log averaged spectrum residual metric to obtain a coarse blur map, then an iterative updating

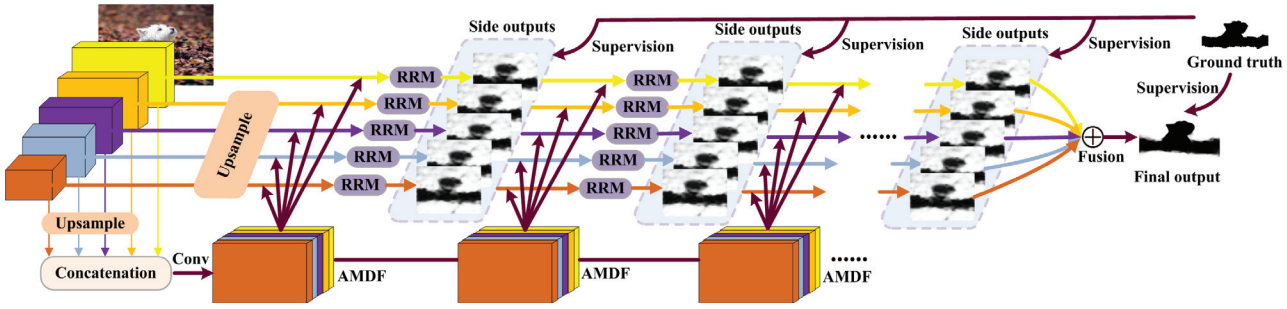


Figure 2: The pipeline of our R^{2}MRF. The lavender block represents our proposed RRM module. Given an input image, we first extract its multi-scale features by using the basic ResNeXt network. For each feature extracting layer, we construct a residual feature leaning and refining branch by embedding multiple RRMs into it.

mechanism is proposed to refine the blur map from coarse to fine based on the intrinsic relevance of similar neighbor image regions. Golestaneh and Karam (Alireza Golestaneh and Karam 2017) proposed to detect defocus blur maps based on a novel high-frequency multiscale fusion and sort transform of gradient magnitudes. Based on the maximum ranks of the corresponding local patches with different orientations in gradient domain, Xu et al. (Xu, Quan, and Ji 2017) presented a fast yet effective approach to estimate the spatially varying amounts of defocus blur at edge locations, then the complete defocus map is generated by a standard propagation procedure.

Although previous hand-crafted methods have earned great success for defocus blur region detection, they can only work well for images with simple structures but are not robust enough for complex scenes. Therefore, extracting high level and more discriminative features is necessary.

Deep Learning based Methods

Due to their high level feature extraction and learning power, deep CNNs based methods have refreshed the records of many computer vision tasks (Simonyan and Zisserman 2015), including defocus blur detection (Park et al. 2017; Zhao et al. 2018). In (Park et al. 2017), high-dimensional deep features are first extracted by using a CNN-based model, then these features and traditional hand-crafted features are concatenated together and fed into a fully connected neural network classifier for defocus degree determination. Purohit et al. (Purohit, Shah, and Rajagopalan 2018) proposed to train two sub-networks which aim to learn global context and local features respectively, then the pixel-level probabilities estimated by two networks are aggregated and feed into a Markov Random Field based framework for blur regions segmentation. Zhang et al. (Zhang et al. 2018a) proposed a dilated fully convolutional neural network with pyramid pooling and boundary refinement layers to generate blur response maps. Considering that the degree of defocus blur is sensitive to scales, Zhao et al. (Zhao et al. 2018) proposed a multi-stream bottom-top-bottom fully convolutional network (BTBNet) which integrates low-level cues and high-level semantic information for defocus blur detection. Since it uses two streams, i.e., a forward stream and a backward stream, to integrate features from different levels for multiple image scales, the computational complexity for both network training and testing of BTBNet is high. Meanwhile, some low-contrast focal areas still cannot be differentiated. Tang et al. (Tang et al. 2019) proposed a defocus blur detection method via recurrently fusing and refining multi-scale deep features and obtained state-of-the-art results. Zhao et al. (Zhao et al. 2019) broke the defocus blur detection into multiple smaller defocus blur detectors and proposed a cross-ensemble network to cancel out the estimation errors of different detectors.

Proposed R^{2}MRF

In this work, we aim to develop an efficient deep neural network for defocus blur detection, which takes a still image as input and output a defocus blur detection map with the same resolution as the input image. Figure 2 shows the entire architecture of our proposed network.

For an effective defocus blur detection deep neural network, it should be capable of extracting both low-level cues and high-level semantic information for generate the final accurate detected defocus blur map. On the one hand, the low-level features can help refine the sparse and irregular detection regions; On the other hand, the high-level semantic features can serve to locate the blurry regions as well as suppress background clutter. In addition, there are often some smooth in-focus regions within an object, which causes these regions to be mistakenly detected as blurry ones due to the lack of rich structure. To this end, the high-level semantic information produced by deep layers should be utilized to avoid this problem. Furthermore, since the defocus degree is sensitive to image scales, the network should be able to make use of multi-scale features for improving the final results. Finally, the network should be easily to be fine-tuned because there are no sufficient labelled defocus blur images for training such a deep network.

Specifically, we choose the ResNet structure (He et al. 2016) as our backbone feature extraction network and use the pre-trained ResNeXt model to initialize our network, which produces five basic feature extraction layers, i.e., conv1, conv2_x, conv3_x, conv4_x, conv5_x. Firstly, we use the backbone network to extract a set of hierarchical features which encode both the low-level details and high-level semantic information with different scales of an image. On

the one hand, since a series of spatial pooling and convolution operations progressively downsample the resolution of the initial image, the fine details of image structure are inevitably damaged, which is harmful for densely separating in-focus and out-of-focus image regions. On the other hand, the high-level semantic features extracted by deep layers can help to locate the defocus blur regions. Therefore, how to exploit the complementary information of features extracted from different layers to improve the final results is critical. Since the residual learning has exhibited better performance than common plain network in many computer vision tasks, we design an RRM to learn the residual between the intermediate prediction and the ground truth and construct a recurrent residual refinement branch for each layer by embedding multiple RRMs into it. In order to sufficiently utilize the complementary information of features extracted from different layers of the backbone network, we aggregate the multi-level features and use it to refine the residual learning process of each RRM. Particularly, we generate Aggregated Multi-level Deep Features (AMDF) by upsampling the feature maps of the last four layers to the size of the feature maps extracted from the first layer, concatenating them together, and applying a convolution operation to merge these feature maps and reduce the feature dimensions, this process can be mathematically formulated as:

$$AMDF = \sigma(\mathbf{W} * (Cat(\mathbf{F}_1, \mathbf{F}_2^{up}, \cdots, \mathbf{F}_5^{up})) + \mathbf{b}), \quad (1)$$

where $\mathbf{F}_1 \in W_1 \times H_1 \times C_1$ is the feature map of the first layer of ResNeXt, $\mathbf{F}_i^{up}|_{i=2,3,4,5} \in W_1 \times H_1 \times C_i$ denotes the enlarged feature map from the i-th layer with C_i channels; $W_1 \times H_1$ is the resolution of the feature map of the first layer; *Cat* represents the concatenation operation across channels; $*$ represents convolution operation; $\mathbf{W}$ and $\mathbf{b}$ are the weights and bias of the convolution need to be learned during training; σ is the activation function and we use ReLU (Krizhevsky, Sutskever, and Hinton 2012) in our work. Hu et al. (Hu et al. 2018) used a similar feature aggregation strategy for salient object detection. However, they didn't focus the scale information which is very important for defocus blue detection.

By using Eq.(1), we can integrate multi-level features to enhance the capability for separating defocus regions from in-focus regions. The shallow layers are more effective to extract subtly fine features to represent delicate image structures, which can improve the accuracy of the defocus blur detection map. Meanwhile, the deep layers can capture high-level semantic features, which well describe the attributes of image contents so as to distinguish smooth in-focus regions from blurry ones as well as locate defocus regions and suppress background clutter. Both of them are utilized to boost the blur detection results.

Since the defocus degree is sensitise to image scales while different layers just extract features with different scales, we construct a recurrent feature refinement branch for each layer and obtain the final detection results by fusing the side output of each branch. Instead of using common plain network for feature learning, we design an RRM to learn the residual between the intermediate prediction and the ground truth by considering that the residual learning has exhibited better performance than traditional plain network in many computer vision tasks. In order to capture the scale information of original input image, the learned residual output of each RRM from each layer is added to the input of RRM of its previous layer in a hierarchical manner at each recurrent step. Multiple RRMs are embedded into each network branch in a recurrent manner for feature refining and the generated AMDF is used to refine the residual learning process in each RRM.

Residual Refinement Module

Compared to common plain network, residual learning has achieved better performance in many computer vision tasks, such as image classification (He et al. 2016) and super-resolution (Zhang et al. 2018b), Therefore, in this work, we design a residual refinement module (RRM) at each recurrent step to correct prediction errors. Figure 3 briefly demonstrates the architecture of the proposed RRM. Specifically, RRM takes the aggregated multi-level deep features (AMDF), the predicted result at the previous step and the residual output of the next layer at the same recurrent step as inputs, and outputs a refined prediction by adding the previous result with a learned residual map. For each side output branch at the l-th layer, the residual map at the t-th recurrent step ($\mathbf{R}_t^l$) can be calculated as:

$$\mathbf{R}_t^l = \begin{Bmatrix} \mathcal{F}(\mathbf{F}_1, AMDF), & t{=}1 \\ \mathcal{F}(Cat(\mathbf{O}_{t-1}^l, AMDF)), & t{>}1 \end{Bmatrix}, \ l = 5, \quad (2)$$

and

$$\mathbf{R}_t^l = \begin{Bmatrix} \mathcal{F}(Cat(\mathbf{F}_1, \mathbf{O}_t^{l+1}, AMDF)), & t{=}1 \\ \mathcal{F}(Cat(\mathbf{O}_{t-1}^l, \mathbf{O}_t^{l+1}, AMDF)), & t{>}1 \end{Bmatrix}, \ l < 5, \quad (3)$$

where $\mathbf{O}_t^l$ is the output of the t-th recurrent step at the l-th layer and $\mathcal{F}$ is a mapping function which consists of a series of convolution an ReLU operations. Then the output of current recurrent step at the l-th layer can be obtained by adding $\mathbf{R}_t^l$ with $\mathbf{O}_{t-1}^l$ in an element-wise manner, which is computed as:

$$\mathbf{O}_t^l = \mathbf{R}_t^l \oplus \mathbf{O}_{t-1}^l. \quad (4)$$

In order to further exploit the image features at different scales at a fine-grained level, we construct a dual path network in each RRM and different paths use different convolutional kernel (as shown on the top of Figure 3). In such a manner, the information between the two paths can be shared with each other so that able to detect the image features at different scales. In addition, in order to improve the prediction accuracy of each intermediate output at each recurrent step, the supervision signal (Xie and Tu 2015) is imposed to each RRM for computing the loss between the ground truth and each intermediate prediction during the training process.

There are three advantages by proposing and embedding RRM into R^{2}MRF. Firstly, better prediction results can be obtained than using traditional plain network. Secondly, the generated AMDF which can capture the complementary information of deep features extracted from different layers is easily integrated into each RRM to refine the residual learning process. Thirdly, since learning residual is much easier,

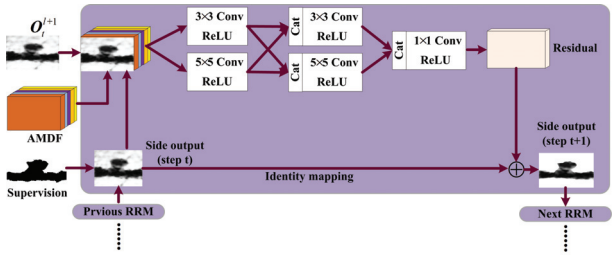


Figure 3: The architecture of the proposed residual refinement module (RRM) at the t-th recurrent step of the l-th layer.

the proposed RRM can ease the learning task with a faster convergence at early stages, and both the training error and time cost can be effectively reduced. We will validate these advantages in the experiments section.

Defocus Maps Fusing

Since the degree of defocus blur is sensitive to image scales, we need to capture multi-scale information for improving final defocus blur detection results. In (Zhao et al. 2018), Zhao et al. proposed to use a multi-stream strategy to fuse the detection results from different image scales. However, this inevitably increase the computational burden of the whole network. In this work, by considering that different layers of neural network just extract image features with different scales, we fuse the side outputs of all the recurrent branches at the last step to generate the final defocus blur map.

Specifically, we first concatenate the side output results predicted from the 5 different recurrent branches, then a convolution layer with a ReLU activation is imposed on the concatenated maps to obtain the final output defocus blur map $\mathbf{B}$, which can be formulated as:

$$\mathbf{B} = \sigma(\mathbf{W}_B * Cat(\mathbf{O}_t^1, \mathbf{O}_t^2, \cdots, \mathbf{O}_t^5) + \mathbf{b}_B), \quad (5)$$

where $\mathbf{O}_t^i$ denotes the intermediate prediction of the t-th step in the i-th recurrent branch; $\mathbf{W}_B$ and $\mathbf{b}_B$ are the weight and bias of the convolution layer on the concatenated side outputs to learn their relationship.

Model Training and Testing

Our network uses the ResNet architecture (He et al. 2016) as backbone and we implement it by using the Pytorch framework. The well trained ResNeXt network on ImageNet (Xie et al. 2016) is used to initialize the parameters of feature extraction network. Therefore, we have five feature extraction layers including conv1, conv2_x, conv3_x, conv4_x, and conv5_x (He et al. 2016; Xie et al. 2016). More details will be found in the released code.

Training: The cross-entropy loss is used for each intermediate output of our network during the training process. For the i-th side output branch at the t-th recurrent step, the pixel-wise cross entropy loss between $\mathbf{O}_t^i$ and the ground truth blur mask $\mathbf{G}$ is calculated as:

$$L_t^i(\boldsymbol{\theta}) = -\sum_{x=1}^{W}\sum_{y=1}^{H}\sum_{l\in\{0,1\}} \begin{Bmatrix} \log \Pr(\mathbf{O}_t^i(x,y)=l|\boldsymbol{\theta}) \\ \cdot \mathbf{1}(\mathbf{G}(x,y)=l) \end{Bmatrix}, \quad (6)$$

where $\mathbf{1}(\cdot)$ is the indicator function. The notation $l \in \{0,1\}$ indicates the out-of-focus or in-focus label of the pixel at location (x,y) and $Pr(\mathbf{O}_t^i(x,y) = l|\boldsymbol{\theta})$ represents its corresponding probability of being predicted as blurry pixel or not. $\boldsymbol{\theta}$ denotes the parameters of all network layers.

Based on Eq. (6), the final loss function is defined as the loss summation of all intermediate predictions:

$$L = \lambda_f L_f + \sum_{i=1}^{5}\sum_{t=1}^{T} \lambda_t^i L_t^i(\boldsymbol{\theta}), \quad (7)$$

where L_f is loss for the final fusion layer; L_f is the weight for the fusion layer and λ_t^i represents the weight of the i-th side output branch at the t-th recurrent step. In our experiment, we empirically set all the weights to 1.

Our R^{2}MRF is initialized by the well trained ResNeXt network on ImageNet (Xie et al. 2016). For fair comparison in the experiments, we fine tune R^{2}MRF on part of Shi et al.'s public blurred image dataset (Shi, Xu, and Jia 2014) as done in the work of BTBNet (Zhao et al. 2018). The dataset consists of 1,000 blurred images and their manually annotated ground truths. 704 of these images are partially defocus blurred and the rest 296 ones are motion blurred. We divide the 704 defocus blurred images into two parts, i.e., 604 for training and the remaining 100 ones for testing. Since the number of training images is not enough to train a deep neural network, we perform data augmentation by randomly rotating, resizing and horizontally flipping all of the images and their corresponding ground truths, and finally the training set is enlarged to 9,664 images. We train our model on a machine equipped with an Intel 3.4GHz CPU with 32G memory and a Nvidia GTX 1080Ti GPU. We optimize the whole network by using Stochastic gradient descent (SGD) algorithm with the momentum of 0.9 and the weight decay of 0.0005. The learning rate is adjusted by the "poly" policy with the power of 0.9. The training batch size is set to 4 and the whole learning process stops after 6k iterations. The training process is completed after approximately 0.75 hours.

Inference: In the testing phase, for each input image, we feed it into our network and obtain the final defocus blur map.

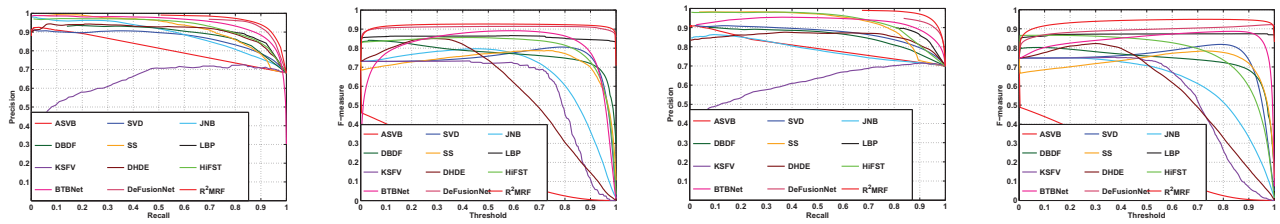
Experiments

Datasets

Similar to previous works, two datasets are used in our experiments for evaluating the performance of our proposed network, they are: **Shi et al.'s dataset** (Shi, Xu, and Jia 2014), which contains the rest 100 defocus blurred images as mentioned above. **DUT** (Zhao et al. 2018), which is a new defocus blur detection dataset which consists of 500 images with pixel-wise annotations. This dataset is very challenging since numerous images contain homogeneous regions, low contrast focal regions and background clutter.

Table 1: Quantitative comparison of F-measure and MAE scores. The best two results are shown in **red** and blue colors, respectively.

Datasets	Metric	ASVB	SVD	JNB	DBDF	SS	LBP	KSFV	DHDE	HiFST	BTBNet	DeFusionNET	R^{2}MRF
Shi et al.'s dataset	F_β	0.731	0.806	0.797	0.841	0.787	0.866	0.733	0.850	0.856	0.892	0.917	**0.927**
	MAE	0.636	0.301	0.355	0.323	0.298	0.186	0.380	0.390	0.232	**0.105**	0.116	0.119
DUT	F_β	0.747	0.818	0.748	0.802	0.784	0.874	0.751	0.823	0.866	0.887	0.922	**0.950**
	MAE	0.651	0.301	0.424	0.369	0.296	0.173	0.399	0.408	0.302	0.190	0.115	**0.088**



(a) (b) (c) (d)

Figure 4: Comparison of precision-recall curves and F-measure curves of different methods on Shi et al.'s dataset ((a) and (b)), and DUT dataset ((c) and (d)).

Evaluation Metrics

Four widely-used metrics are used to quantitatively evaluate the performance of the proposed model: precision-recall (PR) curves, F-measure curves, F-measure scores (F_β) and mean absolute error (MAE) scores.

Comparison with the state-of-the-art methods

We compare our method against other 11 state-of-the-art algorithms, including 3 deep learning-based methods, i..e, DHDE (Park et al. 2017), BTBNet (Zhao et al. 2018) and DeFusionNET (Tang et al. 2019), and 8 classic defocus blur detection methods, including ASVB (Chakrabarti, Zickler, and Freeman 2010), SVD (Su, Lu, and Tan 2011), JNB (Shi, Xu, and Jia 2015), DBDF (Shi, Xu, and Jia 2014), SS (Tang et al. 2016), LBP (Yi and Eramian 2016), KSFV (Pang et al. 2016) and HiFST (Alireza Golestaneh and Karam 2017). For all of these methods except BTBNet, we use the authors' original implementations with recommended parameters. As to BTBNet, we directly download the results from the authors' project website since they have not released their implementation.

Quantitative Comparison. Table 1 presents the compared results of MAE and F-measure scores. It is observed that our method consistently performs favourably against other methods on the two datasets, which indicates the superiority of our R^{2}MRF over other approaches. In Figure 4, we plot the PR curves and F-measure curves of different methods on different datasets. From the results, we observe that our method also consistently outperforms other counterparts.

Qualitative Comparison. Due to the page limitation, we show the visual comparison of our method and other ones in the supplementary. As can be seen from the results, our method generates more accurate defocus blur maps when the input image contains in-focus smooth regions and background clutter. In addition, the boundary information of the in-focus objects can be well preserved in our results. It should be noted that when the background is in-focus and foreground regions are blurred, our R^{2}MRF also works well.

Running Efficiency Comparison. In addition to the appealing results, our proposed R^{2}MRF is also efficient for both training and testing. The whole training process of our R^{2}MRF takes only about 0.75 hours. As to the testing phase, we use only one Nvidia GTX 1080Ti GPU. The average running time for an image of different methods on the two different datasets are shown in Table 2. As can be seen, when our R^{2}MRF is well trained, it is faster than all of other methods for detecting the defocus blur regions from an input image. Although the implementation platforms of the compared methods are different (i.e., traditional hand-crafted features based methods are implemented on CPU, while deep learning based methods are implemented on GPU), R^{2}MRF is significantly faster than other compared methods including two deep learning based ones. As to BTBNet, we cannot evaluate its running time since its implementation has not been released. However, as claimed in the authors' paper, nearly 5 days needed for training BTBNet and approximately 25 seconds is needed to generate the defocus blur map for a testing image with 320 $\times$ 320 pixels. By contrast, our R^{2}MRF takes only about 0.75 hours for training by using the same training dataset and platform as BTBNet. As to testing, we also use only one Nvidia GTX 1080Ti GPU and the average running time on the two datasets is also much less than other methods (see Table 2).

Ablation Analysis

Effectiveness of RRM. In order to validate the efficacy of the proposed RRM, we remove all of the RRMs in R^{2}MRF and directly refine the intermediate side output without residual learning (denoted as R^{2}MRF_no_RRM). The F-measure and MAE scores of R^{2}MRF_no_RRM on

Table 2: Average running time (seconds) for an image of different methods on different datasets.

	Methods	ASVB	SVD	JNB	DBDF	SS	LBP	KSFV	DHDE	HiFST	BTBNet	DeFusionNET	R^{2}MRF
Datasets	Shi et al.'s dataset	2.04	21.09	11.47	214.83	2.76	57.34	32.748	47.06	2576.24	–	0.103	0.096
	DUT	1.59	10.91	5.12	110.37	1.20	30.38	20.139	21.51	1169.57	–	0.087	0.061

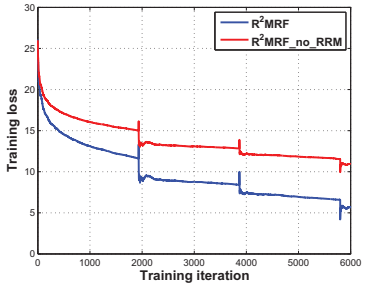


Figure 5: The training loss of R^{2}MRF with/without RRM.

the two datasets are shown in Table 3. As can be seen, our R^{2}MRF with RRM module performs significantly better than R^{2}MRF_no_RRM, which shows that R^{2}MRF with residual learning is superior to the case without residual learning, demonstrating the effectiveness of the proposed RRM. In addition, R^{2}MRF_no_RRM also performs better than other previous methods, which also validates the efficacy of the AMDF for feature refining. We also plot the training loss of R^{2}MRF with/without RRM in Figure 5. As shown in Figure 5, the residual learning can ease the optimization process with a faster convergence at early stages as well as reduce the training error over directly refining the intermediate side outputs.

Effectiveness of the Times of Recurrent Steps. In our R^{2}MRF, we refine the features of each side output branch in a recurrent manner, the feature maps can be improved step by step. In order to validate whether the features can be improved in a recurrent manner, we report the F-measure and MAE scores by using different times of recurrent step in Table 3. As can be seen, the more times of recurrent step, the better results can be obtained. In addition, it should be noted that R^{2}MRF can obtain relatively stable results after 6 times of recurrent step. Therefore, we empirically set 6 times of recurrent step in our experiments for the tradeoff between effectiveness and efficiency. Some visual results with different times of recurrent step can be found in the supplementary file.

Effectiveness of the Final Side Outputs Fusion. By considering that the degree of defocus in an image is sensitive to image scales, we fuse the side outputs of different recurrent branches at the last step to form the final result. We also perform ablation experiments to evaluate the effectiveness of the final fusing step. The final outputs of all the recurrent branches are represented as R^{2}MRF_O1, R^{2}MRF_O2, R^{2}MRF_O3, R^{2}MRF_O4, and R^{2}MRF_O5. We also show the F-measure, MAE scores in Table 3. It can be observed that the fusing mechanism effectively improves the final results. We also give some visual results of different side outputs in the supplementary file.

Table 3: Ablation analysis using F-measure and MAE scores.

Methods	Shi et al.'s dataset		DUT	
	F_β	MAE	F_β	MAE
R^{2}MRF	0.927	0.104	0.950	0.088
R^{2}MRF_no_RRM	0.905	0.165	0.918	0.126
R^{2}MRF_Step_1	0.907	0.163	0.921	0.124
R^{2}MRF_Step_2	0.913	0.155	0.924	0.115
R^{2}MRF_Step_3	0.918	0.148	0.931	0.107
R^{2}MRF_Step_4	0.921	0.140	0.936	0.102
R^{2}MRF_Step_5	0.926	0.126	0.947	0.090
R^{2}MRF_Step_6	0.927	0.119	0.950	0.088
R^{2}MRF_Step_7	0.928	0.119	0.952	0.088
R^{2}MRF_Step_8	0.928	0.118	0.952	0.087
R^{2}MRF_O1	0.793	0.216	0.834	0.184
R^{2}MRF_O2	0.893	0.142	0.873	0.181
R^{2}MRF_O3	0.924	0.135	0.936	0.092
R^{2}MRF_O4	0.921	0.136	0.927	0.105
R^{2}MRF_O5	0.915	0.140	0.932	0.113
R^{2}MRF_VGG	0.918	0.128	0.932	0.114
R^{2}MRF_DenseNet	0.920	0.125	0.934	0.112

Effectiveness of Different Backbone Network Architectures. In order to demonstrate the affect of different backbone network architectures on the final results. Another two models (denoted as R^{2}MRF_VGG and R^{2}MRF_DenseNet), which use the pre-trained VGG-16 (Simonyan and Zisserman 2015) and DenseNet-161 (Gao et al. 2017) to replace the ResNeXt, respectively, are used to compare with R^{2}MRF. The results shown in Table 3 indicate that R^{2}MRF equipped with ResNeXt has a better performance.

Conclusions

In this work, we propose a deep convolutional network for efficient and accurate defocus blur detection via recurrently refine multi-layer residual features (R^{2}MRF). A residual refinement module is designed and embedded into different recurrent feature refining branches for residual learning. In order to capture the scale information of input image, the learned residual output of each residual refinement module from each layer is added to the input of residual refinement module of its previous layer in a hierarchical manner at each recurrent step, and we finally fuse the side output of each branch to obtain the final blur detection map. Extensive experiments with ablation studies are conducted to demonstrate the superiority of R^{2}MRF in both accuracy and

efficiency.

Acknowledgments

This work was supported in part by NSFC (NO. 61701451, 61773392, 61872371 and 61922088), and in part by the Fundamental Research Funds for the Central Universities, China University of Geosciences (Wuhan) under Grant CUG170654. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan V GPU used for this research. Xinwang Liu is the corresponding author of this paper.

References

Alireza Golestaneh, S., and Karam, L. J. 2017. Spatially-varying blur detection based on multiscale fused and sorted transform coefficients of gradient magnitudes. In *CVPR*, 5800–5809.

Bae, S., and Durand, F. 2007. Defocus magnification. *Computer Graphics Forum* 26(3):571–579.

Chakrabarti, A.; Zickler, T.; and Freeman, W. T. 2010. Analyzing spatially-varying blur. In *CVPR*, 2512–2519.

Couzinie-Devy, F.; Sun, J.; Alahari, K.; and Ponce, J. 2013. Learning to estimate and remove non-uniform image blur. In *CVPR*, 1075–1082.

Elder, J. H., and Zucker, S. W. 1998. Local scale control for edge detection and blur estimation. *IEEE TPAMI* 20(7):699–716.

Gao, H.; Zhuang, L.; Maaten, L. V. D.; and Weinberger, K. Q. 2017. Densely connected convolutional networks. In *CVPR*, 4700–4708.

He, K.; Zhang, X.; Ren, S.; and Jian, S. 2016. Deep residual learning for image recognition. In *CVPR*, 770–778.

Hu, X.; Zhu, L.; Qin, J.; Fu, C.; and Heng, P. 2018. Recurrently aggregating deep features for salient object detection. In *AAAI*, 6943–6950.

Jiang, P.; Ling, H.; Yu, J.; and Peng, J. 2013. Salient region detection by ufo: Uniqueness, focusness and objectness. In *ICCV*, 1976–1983.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *NIPS*, 1097–1105.

Liu, R.; Li, Z.; and Jia, J. 2008. Image partial blur detection and classification. In *CVPR*, 1–8.

Pang, Y.; Zhu, H.; Li, X.; and Li, X. 2015. Classifying discriminative features for blur detection. *IEEE TCyb* 46(10):2220–2227.

Pang, Y.; Zhu, H.; Li, X.; and Li, X. 2016. Classifying discriminative features for blur detection. *IEEE TCyb* 46(10):2220–2227.

Park, J.; Tai, Y. W.; Cho, D.; and Kweon, I. S. 2017. A unified approach of multi-scale deep and hand-crafted features for defocus estimation. In *CVPR*, 2760–2769.

Purohit, K.; Shah, A. B.; and Rajagopalan, A. 2018. Learning based single image blur detection and segmentation. In *ICIP*, 2202–2206. IEEE.

Saad, E., and Hirakawa, K. 2016. Defocus blur-invariant scale-space feature extractions. *TIP* 25(7):3141–3156.

Shi, J.; Xu, L.; and Jia, J. 2014. Discriminative blur detection features. In *CVPR*, 2965–2972.

Shi, J.; Xu, L.; and Jia, J. 2015. Just noticeable defocus blur detection and estimation. In *CVPR*, 657–665.

Simonyan, K., and Zisserman, A. 2015. Very deep convolutional networks for large-scale image recognition. In *ICRL*.

Su, B.; Lu, S.; and Tan, C. L. 2011. Blurred image region detection and classification. In *ACMM*, 1397–1400.

Tai, Y.-W., and Brown, M. S. 2009. Single image defocus map estimation using local contrast prior. In *ICIP*, 1797–1800. IEEE.

Tang, C.; Wu, J.; Hou, Y.; Wang, P.; and Li, W. 2016. A spectral and spatial approach of coarse-to-fine blurred image region detection. *IEEE SPL* 23(11):1652–1656.

Tang, C.; Zhu, X.; Liu, X.; Wang, L.; and Albert, Z. 2019. Defusionnet: Defocus blur detection via recurrently fusing and refining multi-scale deep features. In *CVPR*, 2700–2709.

Tang, C.; Hou, C.; and Song, Z. 2013. Defocus map estimation from a single image via spectrum contrast. *Optics letters* 38(10):1706–1708.

Vu, C. T.; Phan, T. D.; and Chandler, D. M. 2012. s_3: A spectral and spatial measure of local perceived sharpness in natural images. *IEEE TIP* 21(3):934.

Wang, X.; Tian, B.; Liang, C.; and Shi, D. 2008. Blind image quality assessment for measuring image blur. In *CISP*, 467–470. IEEE.

Xie, S., and Tu, Z. 2015. Holistically-nested edge detection. *IJCV* 125(1-3):3–18.

Xie, S.; Girshick, R.; Dollar, P.; Tu, Z.; and He, K. 2016. Aggregated residual transformations for deep neural networks. In *CVPR*, 1492–1500.

Xu, G.; Quan, Y.; and Ji, H. 2017. Estimating defocus blur via rank of local patches. In *CVPR*, 22–29.

Yan, R., and Shao, L. 2016. Blind image blur estimation via deep learning. *IEEE TIP* 25(4):1910–1921.

Yi, X., and Eramian, M. 2016. Lbp-based segmentation of defocus blur. *IEEE TIP* 25(4):1626–1638.

Zhang, W., and Cham, W.-K. 2009. Single image focus editing. In *ICCVW*, 1947–1954. IEEE.

Zhang, W., and Cham, W.-K. 2012. Single-image refocusing and defocusing. *TIP* 21(2):873–882.

Zhang, Y., and Hirakawa, K. 2013. Blur processing using double discrete wavelet transform. In *CVPR*, 1091–1098.

Zhang, S.; Shen, X.; Lin, Z.; Mech, R.; Costeira, J. P.; and Moura, J. M. 2018a. Learning to understand image blur. In *CVPR*, 6586–6595.

Zhang, Y.; Tian, Y.; Kong, Y.; Zhong, B.; and Fu, Y. 2018b. Residual dense network for image super-resolution. In *CVPR*, 2472–2481.

Zhao, W.; Zhao, F.; Wang, D.; and Lu, H. 2018. Defocus blur detection via multi-stream bottom-top-bottom fully convolutional network. In *CVPR*, 3080–3088.

Zhao, W.; Zheng, B.; Lin, Q.; and Lu, H. 2019. Enhancing diversity of defocus blur detectors via cross-ensemble network. In *CVPR*, 8905–8913.

Zhu, X.; Cohen, S.; Schiller, S.; and Milanfar, P. 2013. Estimating spatially varying defocus blur from a single image. *IEEE TIP* 22(12):4879–4891.

Zhuo, S., and Sim, T. 2011. Defocus map estimation from a single image. *PR* 44(9):1852–1858.