Deep Structure-Revealed Network for Texture Recognition

Wei Zhai*, Yang Cao*, Zheng-Jun Zha†, HaiYong Xie , Feng Wu
University of Science and Technology of China
wzhai056@mail.ustc.edu.cn, forrest,zhazj,hxie,fengwu@ustc.edu.cn

Abstract

Texture recognition is a challenging visual task since various primitives along with their arrangements can be recognized from a same texture image when perceiving with different contexts. Some recent work building on CNNs exploits orderless aggregating to provide invariance to spatial arrangements. However, these methods ignore the inherent structural property of textures, which is a critical cue for distinguishing and describing texture images in the wild. To address this problem, we propose a novel Deep Structure-Revealed Network (DSR-Net) that leverages spatial dependency among the captured primitives as structural representation for texture recognition. Specifically, a primitive capturing module (PCM) is devised to generate multiple primitives from eight directional spatial contexts, in which deep features are firstly extracted under the constrains of direction map and then encoded based on the similarities of neighborhood. Next, these primitives are associated with a dependence learning module (DLM) to generate structural representation, in which a two-way collaborative relationship strategy is introduced to perceive the spatial dependencies among multiple primitives. At last, the structure-revealed texture representations are integrated with spatial ordered information to achieve real-world texture recognition. Evaluation on the five most challenging texture recognition datasets has demonstrated the superiority of the proposed model against state-of-the-art methods. The structure-revealed performances of DSR-Net are further verified on some extensive experiments, including fine-grained classification and semantic segmentation.

1. Introduction

As texture is the fundamental microstructure of natural images and the preattentive human visual cue for perceiving natural scene, it serves as a significant mid-level feature representation for a wide variety of applications, such as

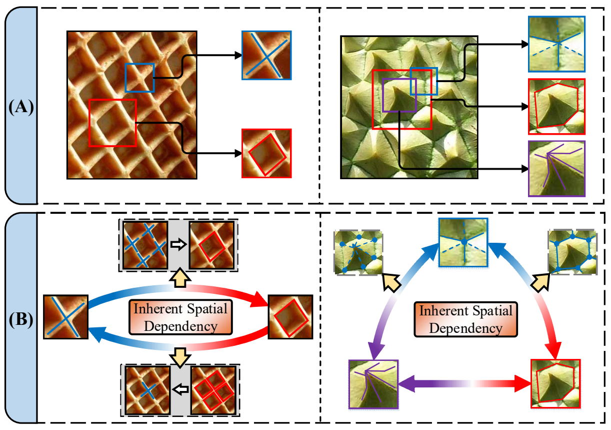


Figure 1. The part (A) shows that textures usually have several probable primitives. The part (B) shows there is an inherent spatial dependency between them several probable primitives.

medical image analysis, industrial visual inspection, image classification/retrieval [14, 27, 21, 20, 34, 30].

Texture refers to spatial organization of a set of basic primitives (i.e. textons) [11], Therefore, a textured region usually conforms to some statistical properties, exhibiting periodically recurrent textons [14]. Some methods exploit the statistical property and incorporate an orderless component to obtain global compact feature representation. Recently, the methods building upon convolutional neural networks (CNNs) inherit this property and utilize an orderless aggregation of local texture features to achieve state-of-the-art performance on texture recognition. For example, Zhang et al. present Deep Texture Encoding Network [33] that integrates dictionary learning and residual encoding into CNN to form an end-to-end texture recognition network. Leveraging DeepTEN as a texture encoding layer, J. Xue et al. [27] further present a Deep Encoding Pooling Network (DEP) to capture the orderless texture details together with local spatial information for ground terrain recognition.

Although the existing texture recognition methods excel at providing invariance to spatial arrangement, they are typically limited in capturing the inherent structure of texture primitives, resulting in incomplete description and in-

*co-first author
†corresponding author

accurate recognition. In this paper, we propose a Deep Structure-Revealed Network (DSR-Net) that performs texture recognition by learning the inherent spatial dependency of multiple primitives. Our method is based on the observation that, though various texture primitives can be perceived with different spatial contexts (e.g. principal directions), there exists a spatial dependency among these primitives, which is invariant to spatial layout and reveals the inherent structural properties of texture. For example, as shown in Fig. 1 (a), multiple primitives can be identified from the same texture image, which will bring interference to texture recognition. However, we can also find that there is inherent spatial dependency among these primitives, which is invariant to spatial transformation and brightness change. It implies that this dependency is robust to the variability in primitive appearance and spatial organizations, and can serve as a structural representation of texture image.

Specifically, to leverage the dependency among primitives for texture recognition, the proposed DSR-Net is devised to consist of two modules, which are primitive capturing module (PCM) and dependence learning module (DLM). PCM first generates deep features under the guidance and constraints of eight principal directional maps, and then encodes them based on the similarities of neighborhood to capture the robust representations for candidate primitives. DLM associates these representations and perceives the spatial dependencies among them by a two-way collaborative relationship strategy. Inspired by the success of DEP method, the dependency-based representations are then integrated with spatial ordered information to achieve texture recognition for images in the wild. The resultant network shows excellent performance not only for texture recognition, but also for general visual tasks, such as semantic segmentation and fine-grained classification.

Our contributions are summarized as follows:

(1) This paper presents a Deep Structure-Revealed Network (DSR-Net), which perceives spatial dependency among multiple texture primitives and leverages it as inherent structural property for texture recognition.

(2) This paper proposes a novel primitives dependency learning method, in which multiple primitives are first generated from eight directional contexts by introducing a similarity encoding scheme, and then associated together to generate structural dependencies by a two-way collaborative relation modeling strategy.

(3) Extension evaluations show that the proposed network not only excels at texture recognition, but also works well on general visual tasks, such as fine-grained classification and semantic segmentation.

2. Related Work

Texture representation is an important area of research in computer vision for potential applications in classification, segmentation and synthesis. Although the traditional goal is to represent texture based on their perceptual differences or material types, texture also has other characteristics that reflect from the structure, such as the inherent structure of primitives and structural dependence among primitives (As shown in Fig. 1). Structural-related characteristics play a crucial role in texture analysis, since they reflect the local primitive characteristics along with its spatial layout.

The research of texture representation is mainly divided into two classes: traditional methods and CNN-based method (deep learning period method). In traditional methods, feature extraction are usually realized by methods like gray level co-occurrence matrix [8], LBP [18], filter banks method [25], SIFT [15] and HOG [6]. To get a compact global representation, some studies such as BoWs [4], VLAD [10], and FV [19] are presented to encode and aggregate local features. Finally, some machine learning methods (e.g. SVM) are used to generate global representations.

In deep learning period, previous methods such as FVCNN [3] combine the deep network feature extraction with the BoWs method, LFV [24] directly use the FV method to encode features extracted by deep network. DeepTEN [33] uses the classic idea of residual dictionary learning to encode features, embedding the coding layer into the network for end-to-end training. To generate invariance to spatial layout, orderless bilinear pooling [13, 5] is introduced to polarize strong response and weak response features. DEP [27] further integrates orderless texture details and local spatial information to achieve recognition for texture in the wild. The advance of these existing methods is the powerful basic feature representation (CNN and SIFT) and the ability to features encoding (FV method).

One of the main challenges of texture recognition is that various primitives along with their arrangements can be recognized from a same texture image when perceiving with different context. To eliminate this variability, most of the existing methods use the orderless component to obtain the invariance for the spatial arrangement of multiple primitives. Unlike them, this paper aims to exploits the spatial dependency among the recognized primitives, and leverage it as inherent structural representation for texture recognition.

3. Deep Structure-Revealed Network

3.1. Architecture

In this paper, we propose a novel Deep Structure-Revealed Network (DSR-Net) to capture and represent the most probable structure of primitives by exploiting their inherent spatial dependency. The overall architecture of our proposed method is illustrated in Fig. 2 (a). Our network consists of two branches: structure revealed branch and spatial ordered Branch. Similar to [33, 27, 31], we also use

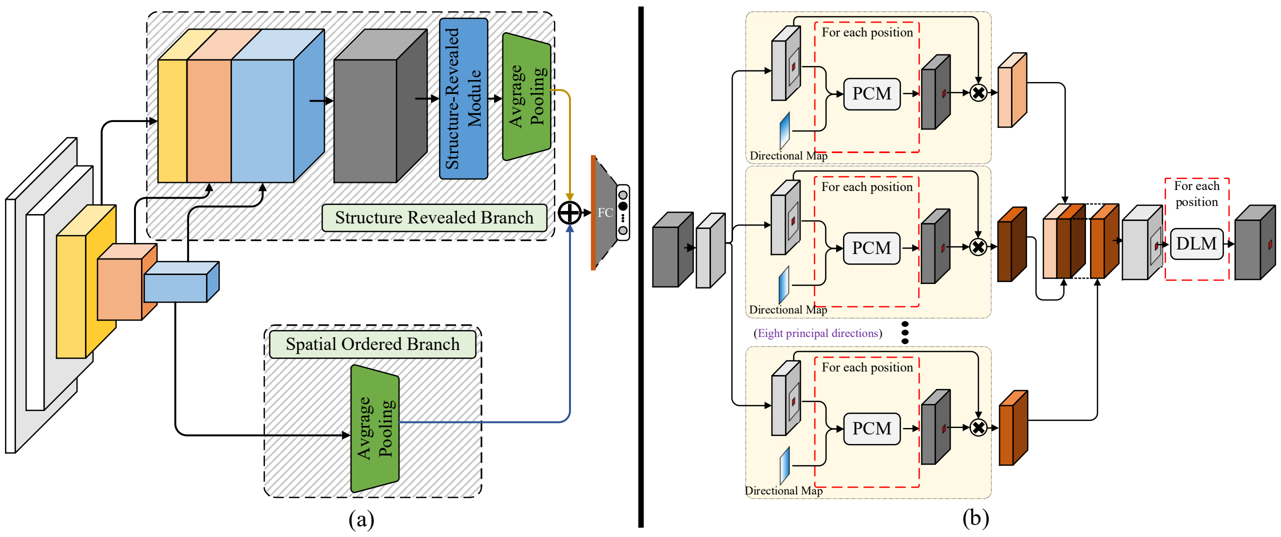


Figure 2. (a) The architecture of our proposed DSR-Net. Our network consists of two branches: Structure Revealed Branch and Spatial Ordered Branch. (b) The details of the Structure-Revealed module.

resnet50 as our feature extractor. To comprehensively capture the local spacial features, we concatenate features of different level together as a feature pool. Different feature maps are then upsampled to obtain the same resolution as the original feature map via bilinear interpolation. To decrease the computation cost, a 1×1 convolution layer is used to reduce the number of channels to 2048. Similar to [31], an adaptive avgrage pooling is exploited to adapt to local features with different sizes, resulting in an orderless aggregation. Texture images in the wild are rarely filled by a single homogeneous surface but instead contain both orderless and ordered components. To address this problem, we refer to the method in DEP [27] and apply a global average pooling operation to balance the orderless representations with spatial ordered information. In the last, to deal with the nature texture without filling all in the image, the structure-revealed features are merged with spatial ordered features through an element-sum operation, just like [27].

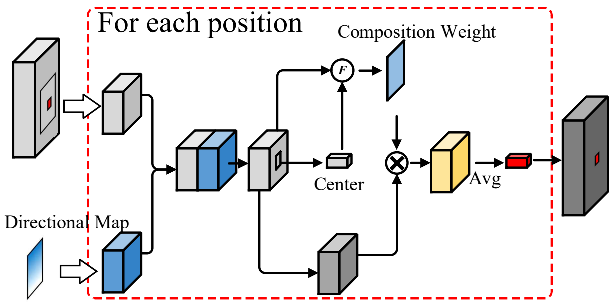


Figure 3. Primitive Capturing Module. We use directional map as directional context guidance to capture candidate primitives and improve the robustness of captured primitives by composition weight. PCM is a new convolutional computing layer. ‘For each position’ means every step in the PCM calculation process.

3.2. Deep Structure-Revealed Module

For DSR module, a primitive capturing module (PCM) is devised to generate the candidate primitives with eight principal directional contexts, in which deep features are firstly extracted under the constraint of directional map and then encoded by the similarities of neighborhood. Next, these candidates are associated by a dependence learning module (DLM) to generate the structural representation, in which a two-way collaborative relation modeling strategy is introduced to perceive the spatial dependencies among candidate primitives. Our proposed Deep Structure-Revealed Module is illustrated in Fig. 2 (b).

Primitive Capturing Module. For primitives, elements in texture images, we need to capture them locally under the guidance of different spatial context. Therefore, a new primitive capturing module is devised to generate candidate primitives with provided spatial context constraints. Specifically, we mainly consider the contexts in eight directions (up, down, left, right, left top, right upper, left bottom, right bottom). A directional map with the same size as the convolution kernel is provided as the local context guidance in the convolution process. In the experiment, to facilitate the network convergence, we apply a final linear scaling of directional map values to make it falls in the range of $[-1, 1]$. However, there is a gap between directional map and original feature space. Therefore, we devise a small convolution network to map the direction information to the original fea-

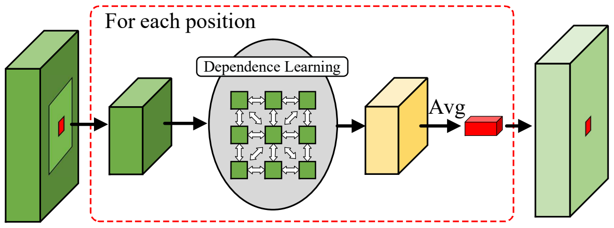


Figure 4. Dependence Learning Module. DLM is a new convolutional computing layer. 'For each position' means every step in the DLM calculation process.

ture space. It can be expressed as follows:

$$x_{\mathbf{p}}^{D} = cat(x_{\mathbf{p}}, f_D(D_{\mathbf{p}})), \tag{1}$$

where cat is the concatenation operation, x is the input feature, D is the directional map, $\mathbf{p} = (w, h)$, $x_{\mathbf{p}}$ is the feature with the size same as kernel, and f_D is a embedding function. To increase robustness to illumination change and spatial distortion, a composition operation referred in [9] is adopted, which is expressed as follows:

$$\begin{aligned} F(x_p^D, &x_{p_{center}}^D) = \\ &sigmoid(\Phi(f_p(x_p^D), f_{p_{center}}(x_{p_{center}}^D))), \end{aligned} \tag{2}$$

where p_{center} is the center of $\mathbf{p}$, $sigmoid$ works as a normalization operation, and Φ is a measure of composability between p and p_{center}. In this paper, $\Phi(p, p_{center}) = -(p - p_{center})^2$. f_p and $f_{p_{center}}$ are the transformation function. The primitives capturing process can be expressed as follows:

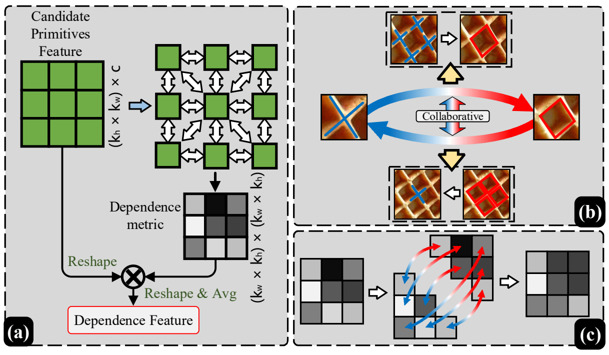
$$y_{primitives} = \frac{1}{k_w \times k_h} \sum F(x_p^D, x_{p_{center}}^D) \cdot x_{\mathbf{p}}, \tag{3}$$

where y is the output of one position in the convolution process, (k_w, k_h) is the size of $\mathbf{p}$. The complete PCM is illustrated in Fig. 3.

Dependence Learning Module. The dependency learning module is used to establish the dependency relationship among multiple candidate primitives in the local scope (as shown in Fig. 4). In dependence learning module, we consider the feature $x_{\mathbf{p}}$ of local region as a set of feature vectors $\mathbf{V} \in \mathbb{R}^{c \times \theta \times 1}$, we first feed it into a transformation to generate two new features $\mathbf{V^1}$ and $\mathbf{V^2}$ respectively, where $\{\mathbf{V^1}, \mathbf{V^2}\} \in \mathbb{R}^{c \times \theta \times 1}$. Then we perform a matrix multiplication between the transpose of $\mathbf{V^2}$ and $\mathbf{V^1}$, and apply a softmax layer to calculate the dependence matrix $\mathbf{R} \in \mathbb{R}^{\theta \times \theta}$:

$$r_{ij} = \frac{exp(V_i^1, V_j^2)}{\sum_{i=1}^{\theta} exp(V_i^1, V_j^2)}, \tag{4}$$

where r_{ij} measures the i^{th} position's impact on j^{th} position, $\theta = k_w \times k_h$ (Fig. 5 (a)). The more similar feature


Candidate Primitives Feature
$(k_h \times k_w) \times c$
Dependence metric
$(k_w \times k_h) \times (k_w \times k_h)$
Reshape
Reshape & Avg
Dependence Feature
(a)
Collaborative
(b)
(c)
Figure 5. (a) The calculation process of the dependence learning module. (b) There is a coupling between the two-way connections of primitives. (c) The implementation process of a two-way collaborative relationship strategy.

representations of the two elements are, the greater the correlation between them is. It can be seen that the current approach is equivalent to establishing a two-way relationship between two elements. Furthermore, to capture the correlation between the two-way relationships as shown in Fig. 5 (b), we propose a two-way collaborative strategy as shown in Fig. 5 (c), to improve the dependence modeling ability by introducing correlations between the two-way relation. The two-way collaborative strategy can be expressed as follows:

$$\bar{r_{ij}} = \frac{exp(r_{ij})}{exp(r_{ij}) \cdot exp(r_{ji})}, \tag{5}$$

$$\bar{r_{ji}} = \frac{exp(r_{ji})}{exp(r_{ij}) \cdot exp(r_{ji})}, \tag{6}$$

where r_{ij} and r_{ji} are a pair of two-way relationship in $\mathbf{R}$, which are processed before the two-way collaborative operation. The $\bar{r_{ij}}$ and $\bar{r_{ji}}$ are a pair of two-way relation, which are processed after the two-way collaborative operation.

Meanwhile, we feed feature $\mathbf{V}$ into a convolution layer to generate a new feature $\mathbf{V^3} \in \mathbb{R}^{c \times \theta \times 1}$. Then we perform a matrix multiplication between $\mathbf{V^3}$ and the transpose of $\mathbf{R}$. Finally, we apply an element-wise sum operation on the features $\mathbf{V}$ to obtain the final output $\mathbf{E} \in \mathbb{R}^{c \times \theta \times 1}$, expressed as follows:

$$\mathbf{E}_i = \sum_{j=1}^{\theta} (r_{ij} \mathbf{V}_j^3) + \mathbf{V}_i, \tag{7}$$

It can be inferred from Eq. (7) that each element at the resulting feature $\mathbf{E}$ is a weighted sum of the features across all elements and original features. Finally, the dependency feature at each local scope can be expressed as follows:

$$y_{dependence} = \frac{1}{\theta} \sum E_i. \tag{8}$$

4. Experiments

In this section, we evaluate the proposed method on five texture/material datasets and then analyze the effectiveness of the method. Furthermore, we also present the testing results of applying our DSR module to fine-grained recognition and semantic segmentation tasks.

4.1. Datasets and implementation

Experimental data. We evaluate our model on five challenging texture/material recognition datasets. Describable Texture Database (DTD) [2] is considered as the most widely used benchmark for texture recongnition which contains 47 texture categories with a total of 5640 images. Flickr Material Dataset (FMD) [23, 22] consists of 10 material categories, each of which contains 100 images. KTH-TISP2b (KTH) [1, 17] dataset contains 11 material categories with a total of 4752 images. Ground Terrain in Outdoor Scenes (GTOS) [28] is a dataset of ground materials in outdoor scene with 40 categories. GTOS-mobile [33] is collected from GTOS dataset by mobile phone, which consists of 31 material classes. For DTD, FMD and KTH datasets, we randomly divide each dataset into 10 splits and report the mean accuracy across splits. For GTOS and GTOS-mobile datasets, the evaluation is based on provided train-test splits. Similar to [33], the result accuracy $mean \pm std\%$ are reported. The results on DTD, FMD, KTH and GTOS datasets are based on 5-time statistics, and the results on GTOS-mobile datasets are averaged over 2 runs.

Implementation details. We implement our model with PyTorch, and two TITAN Xp GPUs are used for training. Similar to [31], Resnet-50 is used as feature extractor, in which *res3*, *res4* and *res5* are used as the source of feature pool, the feature maps channels from feature pool are reduced to 2048 with 1 convolutional layer. We adopt a similar training and data augmentation strategy to [31]. At the testing phase, we use the resolution of 224×224 for all datasets.

4.2. Ablation study

To evaluate DSR-Net, we conduct experiments with several settings, including Feature Pool (FP), Primitives Capturing Module (PCM), Dependence Learning Module (DLM) and Spatial Order (SO).

Primitives Capturing module. In this section, we explore the effectiveness of primitives capturing module for DSR-Net. The primitive capturing module is devised to generate the candidate primitives under the constraints of spatial contexts. As shown in Table 1, it outperforms the baseline with an improvement of $3.1\%/4.7\%$ (Baseline $\rightarrow$ Baseline+PCM) in terms of mean accuracy on DTD and GTOS. We also conducted experiments on the number of directions. In detail, we conducted experiments on two directions (up and down), four directions (up, down, right and left), eight directions (up, down, right, left, up left, down left, up right and down right) and none directions (only Resnet50), respectively. The effectiveness of the number of directions is shown in Table 2. On the whole, the growth rate of the final result decreases with the number of directions increasing. It can be seen that multiple directional context guidances are helpful to extract more complete potential primitives, and it can help successive DLM learn more discriminative representations of structure dependency. We find that 3×3 is the best choice for the size of the kernel in PCM, which may be due to the fact that too large kernel will extract the structure information irrelevant to primitives. For the details on the selection of the kernel size in PCM, please refer to our supplementary materials.

Dependence Learning module. In this part, we study the effect of dependence learning module which is proposed to model local structures and their relationships. To improve dependence modeling, a two-way collaborative rela-

Model	FP	PCM	DLM	SO	DTD	GTOS
Resnet50					$68.9_{\pm 1.2}$	$76.0_{\pm 2.8}$
	✓				$70.2_{\pm 1.3}$	$77.6_{\pm 2.8}$
		✓			$72.0_{\pm 0.9}$	$80.7_{\pm 2.5}$
		✓	✓*		$74.3_{\pm 0.9}$	$83.2_{\pm 2.3}$
	✓	✓	✓*		$75.7_{\pm 0.8}$	$83.6_{\pm 2.3}$
	✓	✓	✓*	✓	$76.3_{\pm 0.7}$	$84.2_{\pm 2.2}$
DSR-Net	✓	✓	✓	✓	$\mathbf{77.6}_{\pm \mathbf{0.6}}$	$\mathbf{85.3}_{\pm \mathbf{2.0}}$

Table 1. Ablation study on DTD and GTOS datasets. 'FP' is Feature Pool. 'PCM' is Primitives Capturing Module. 'DLM' is Dependence Learning Module. 'SO' is Spatial Order. '*' is when 'DLM' doesn't use the two-way collaborative strategy. (Acc %)

Number of directions	DTD	GTOS
$None$ (Baseline)	$68.9_{\pm 1.2}$	$76.0_{\pm 2.8}$
Two	$74.9_{\pm 0.8}$	$83.1_{\pm 2.0}$
$Four$	$76.1_{\pm 0.6}$	$84.2_{\pm 2.0}$
$Eight$	$\mathbf{77.6}_{\pm \mathbf{0.6}}$	$\mathbf{85.3}_{\pm \mathbf{2.0}}$

Table 2. Effectiveness of the number of directions on DTD and GTOS. (Acc %)

Kernel size	DTD	GTOS
3×3	$76.7_{\pm 0.6}$	$84.6_{\pm 2.0}$
5×5	$77.3_{\pm 0.7}$	$\mathbf{85.3}_{\pm \mathbf{2.0}}$
7×7	$\mathbf{77.6}_{\pm \mathbf{0.6}}$	$84.3_{\pm 2.1}$
9×9	$77.1_{\pm 0.6}$	$83.7_{\pm 2.1}$
Full image	$75.7_{\pm 0.7}$	$82.1_{\pm 2.5}$

Table 3. The effectiveness of kernel size in DLM. 'Full image' represents the kernel size is the same size as the input feature maps. (Acc %)

Texture Dataset	Conference	Backbone	DTD		KTH-T2b		FMD		GTOS		GTOS-mobile	
Method			*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*std*
FV-VGGVD [3]	CVPR2015	VGGVD	72.3	1.0	75.4	1.5	79.8	1.8	77.1	−−	−−	−−
FC-VGGVD [3]	CVPR2015	VGGVD	62.9	0.8	81.8	2.5	77.4	1.8	−−	−−	−−	−−
B-CNN [13]	CVPR2016	VGGVD	69.6	0.7	5.1	2.8	77.8	1.9	−−	−−	−−	−−
B-CNN [27]	CVPR2016	Resnet18	−−	−−	−−	−−	−−	−−	−−	−−	75.43	−−
LFV [24]	ICCV2017	VGGVD	73.8	1.0	82.6	2.6	82.1	1.9	−−	−−	−−	−−
FASON [5]	CVPR2017	VGGVD	72.3	0.6	76.5	2.3	−−	−−	−−	−−	−−	−−
DeepTEN [33]	CVPR2017	Resnet50	69.6	−−	82.0	3.3	80.2	0.9	84.5	2.9	−−	−−
DeepTEN [27]	CVPR2017	Resnet18	−−	−−	−−	−−	−−	−−	−−	−−	76.12	−−
DEP [27]	CVPR2018	Resnet50	73.2	−−	−−	−−	−−	−−	−−	−−	−−	−−
DEP [27]	CVPR2018	Resnet18	−−	−−	−−	−−	−−	−−	−−	−−	82.18	−−
MAP-net [31]	ICCV2019	VGGVD	74.1	0.6	82.7	1.5	82.9	0.9	80.8	2.5	82.0	1.6
MAP-net [31]	ICCV2019	Resnet18	69.5	0.8	80.9	1.8	80.8	1.0	80.3	2.6	82.98	1.6
MAP-net [31]	ICCV2019	Resnet50	76.1	0.6	84.5	1.3	85.2	0.7	84.7	2.2	86.64	1.5
DSR-Net (ours)		VGGVD	74.9	0.7	83.5	1.5	84.0	0.8	81.8	2.2	82.94	1.6
DSR-Net (ours)		Resnet18	71.2	0.7	81.8	1.6	81.3	0.8	81.0	2.1	83.65	1.5
DSR-Net (ours)		Resnet50	**77.6**	0.6	**85.9**	1.3	**86.0**	0.8	**85.3**	2.0	**87.03**	1.5

Table 4. Compariing our method (DSR-Net) with state-of-the-arts methods on DTD, KTH-T2b, FMD, GTOS and GTOS-mobile. Here 'VGGVD' is VGG-19. (Acc %)

Method	CUB	Car
Resnet50	85.4	91.7
RAM	86.0	−−
DFL-CNN	87.4	93.1
NTS-Net	87.5	−−
Resnet50 + DSR (ours)	**87.6**	**93.3**

Table 5. Fine-grained recognition results on CUB and Stanford Car196. 'Resnet50 + DSR' is our DSR-Net without spatial ordered pathway. (Acc %)

Method	*pixAcc* %	*mIoU* %
FCN	71.32	29.39
SegNet	71.00	21.64
DilatedNet	74.52	34.90
FCN (Resnet50)	74.57	34.38
EncNet (resnet50)	79.73	41.11
PSP (Resnet50)	80.04	41.68
EncNet (Resnet50) + DSR (ours)	80.13	41.85
PSP (Resnet50) + DSR (ours)	**80.64**	**42.35**

Table 6. Semantic segmentation results on the ADE20K validation set. 'EncNet + DSR' is EncNet with a DSR module. 'PSPNet + DSR' is PSPNet with a DSR module.

tionship strategy is adopted to improve dependence learning. As shown in Table 1, the performance is improved by 2.3%/2.5% (Baseline+PCM $\rightarrow$ Baseline+PCM+DLM*) and 1.3%/1.1% (Baseline+FP+PCM+DLM* $\rightarrow$ DSR-Net). It demonstrates that the two-way collaborative strategy effectively improves the performance of dependence learning module, leading to better performance on perceiving local structural properties. We also conducted experiments to evaluate effectiveness of kernel size in DLM, which determines the scope to establish local structural dependence. We set the kernel size from 3 to full image and show the results in Table 3. For DTD dataset, when kernel size is 7×7, the result is the best. For GTOS dataset, when kernel size is 5×5, the result is the best. We also find that, when kernel size is equal to the size of whole input feature map, the result is not ideal. The reason for this may be that modeling structure dependence on a large scope is inevitably affected by the structure of the non-textured region.

Feature Pool and Spatial Order. Here, we evaluate the effectiveness of feature pool and spatial order. As shown in Table 1, the performance is improved by 1.4%/0.4% (Baseline+PCM+DLM* $\rightarrow$ Baseline+FP+PCM+DLM*). The feature pool provides a better foundation representation to the successive tasks. Furthermore, the performance is improved by 0.6%/0.6% (Baseline+FP+PCM+DLM* $\rightarrow$ Baseline+FP+PCM+DLM*+SO). It shows that the spatial ordered provides important cue for the spatial layout of textured region and the other regions.

4.3. Comparisons against state-of-the-arts.

We compare the performance of our method with several texture/material recognition methods on DTD, KTH-T2b, FMD, GTOS and GTOS-mobile datasets. All the methods are listed as follows: **FV-VGGVD**: **F**isher **V**ector **CNN** [3]. **FC-VGGVD**: General VGG-19 [3]. **B-CNN**: **B**ilinear-CNN [13]. **FASON**: **F**irst **A**nd **S**econd **O**rder information fusion **N**etwork [5]. **DeepTEN**: **Deep** **T**exture **E**ncoding **N**etwork [33]. **LFV**: **L**ocally-Transferred **F**isher **V**ectors [24]. **DEP**: **D**eep **E**ncoding **P**ooling Network [27]. **MAP-**

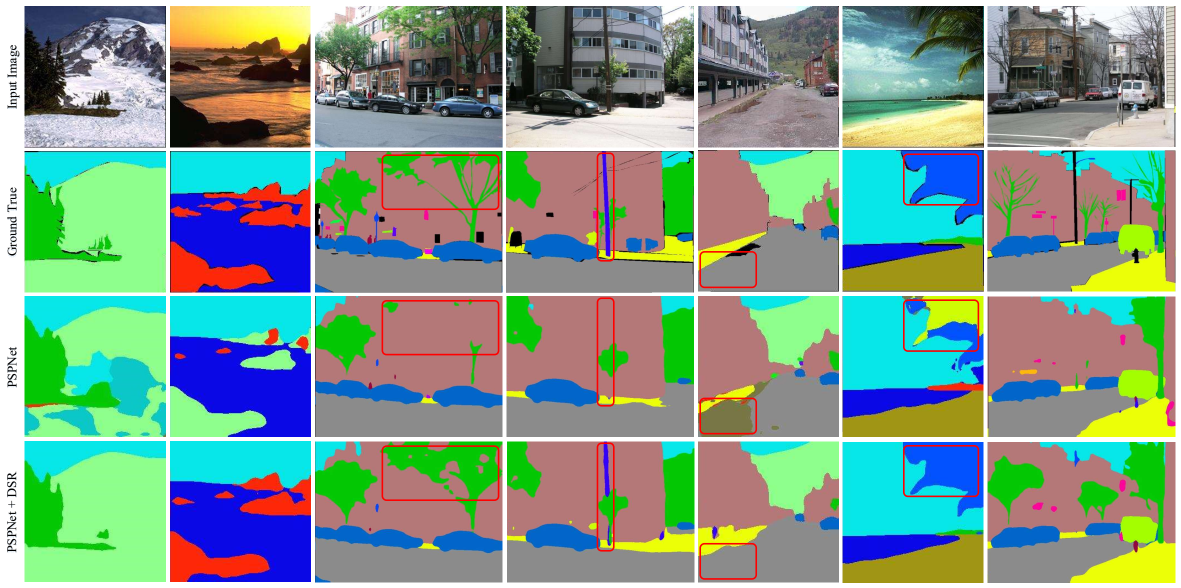


Figure 6. Qualitative segmentation results from the ADE20K [36] validation set for PSPNet with and without the DSR module. Differences are highlighted with red boxes. PSPNet with DSR module produces more accurate predictions.

Net: **M**ultiple-**A**ttribute-**P**erceived **Net**work for real-world texture recognition [31].

For a fair comparison, the texture/material recognition results of other methods are obtained either directly from results provided by the authors, or by generating them using implementations provided by the authors with recommended parameter setting. Moreover, we perform experiment on DSR-Net using different backbones (VGG-19, Resnet-18 and Resnet-50). For Resnet-18 and Resnet-50, *res4* and *res5* are combined as feature pool. The channels of feature maps from feature pool are transformed to 512 and 2048 with 1×1 convolutional layer, respectively. For VGGVD, *conv4_4* and *conv5_4* are combined as feature pool. The channels of feature maps from feature pool are transformed to 512 with 1×1 convolutional layer. Table 4 reports the comparison results, where we can see that our method is $1.5\%/1.4\%/0.8\%/0.6\%/0.49\%$ higher than the state-of-the-art in $mean$ acc for five benchmark datasets. Note that our DSR-Net improves most significantly on DTD in which the appearance and arrangement of primitives vary the most. It demonstrates that the structural properties revealed by DSR-Net can provide robustness to the variability of texture primitives.

4.4. Application.

To demonstrate the structure-revealed ability of our method in general visual tasks, we conduct extend Deep Structure-Revealed module into fine-grained recognition and scene parsing tasks.

Fine-Grained Recognition. Recognizing fine-grained categories by computer vision techniques has attracted extensive attention. Texture can serve as an important mid-level representation for fine-grained recognition. We comprehensively evaluate our DSR module on Caltech-UCSD Birds (CUB-200-2011) [26] and Stanford Cars [12] datasets, which are widely used benchmark for fine-grained image classification. Note that we do not use any bounding box/part annotations in all our experiments. CUB-200-2011 is a bird classification task with 11788 images from 200 wild bird species. The ratio of train data and test data is roughly $1:1$. It is generally considered one of the most competitive datasets since each species has only 30 images for training. Stanford Cars dataset contains 16185 images over 196 classes, and each class has a roughly $50 fi 50$ split. The DSR-Net without Spatial ordered branch is adopted as the model of this experiment. We adopt a similar parameter setting as [29]. Please refer to supplementary material for more implementation details. As shown in Table 5, our proposed method that uses Resnet50 as backbone can achieve better accuracy (87.6% / 93.3%) than main-stream methods. It indacates that the structure-revealed ability of our DSR module is also beneficial for fine-grained recognition.

Scene Parsing. In the next, we extend DSR module into the scene parsing, in which texture and material in the scene are also important cues. Therefore, we introduce the proposed DSR module to capture the structural property of textures for scene parsing. We report the performances on ADE20k [36] validation set. ADE20K dataset is a recent scene parsing benchmark containing denese labels of 150 stuff/object category labels. The dataset includes

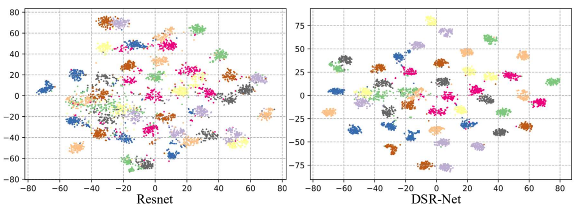


Figure 7. Visualization of features after dimension reduction by t-SNE [16]. We trained DSR-Net and Resnet on DTD dataset and used the vector output from AvgPooling layer as the feature representation.

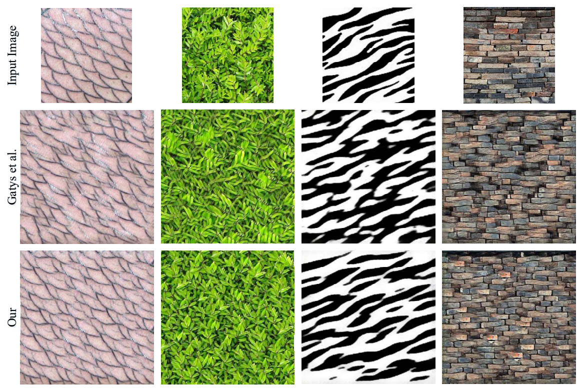


Figure 8. Comparison of texture synthesis results using different models. Our texture templates come from natural images.

$20K/2K/3K$ images for training, and validation. We insert DSR module into the popular model [35, 32] of scene parsing task for this experiment. Please refer to supplementary material for implementation details. We adopt a similar training setting to [35]. As shown in Table 6, the baseline networks (encnet and pspnet) with DSR module have achieved $0.40\%/0.74\%$ and $0.60\%/0.77\%$ improvement compared with the previous. As shown in Fig. 6, after inserting into DSR module, the texture/material regions can be better distinguished as one category, leading to more visual consistent segmentation results.

4.5. Performance Analysis.

To verify the ability of DSR-Net to represent texture features, we compared the distribution of the representations learned from Resnet and DSR-Net in the feature space. Fig. 7 shown the visualization of features after dimension reduction by t-SNE [16]. Compared with Resnet, DSR-Net shows a better feature distribution that exhibits smaller intraclass differences and larger interclass variations. More discriminative representation is obtained by using the inherent structure dependence of texture by our DSR module.

To explicitly demonstrate the performance on structure revealing, we apply our DSR module into texture synthesis. We train our DSR-Net on DTD dataset by using the convolutional layers of VGG-19 as the feature extraction

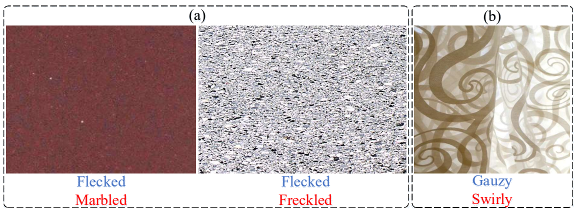


Figure 9. Failure cases of our method. The red font represents the ground truth label, and the blue font represents the prediction result of our method.

component. For fair comparison, we use the convolutional layers of VGG-19 as the feature extraction network in DSR-Net. We train it on DTD dataset. At the test stage, $conv1\_1$, $conv2\_1$, $conv3\_1$ and $conv4\_1$ layers are used for texture synthesis. And we run L-BFGS for 512 iterations in all experiments with learning rate 0.1. Fig. 8 shows the result of texture synthesis comparing with the work of Gatys et al. [7]. On the whole, our synthetic results are more delicate, in which local structure of texture is more complete, and texture arrangement is more regular. It shows that the convolutional layers of VGG-19 learn a detailed representation under the constraint of DSR module.

Here we show some failure cases in Fig. 9. As can be seen, our method fails in some extreme scenes: (a) When the local structure is hard to be identified, the features in deep layers lose the detail information for structural dependence. (b) When there are multiple effective structures in texture images, it is not possible to model structure attributes well without semantic guidance. One feasible solution for these cases is to introduce high-level semantic information as a guidance, just like [31].

5. Conclusion

The variability of multiple primitives bring a great interference to texture recognition. This paper proposes a new solution to utilize spatial dependency among the perceived primitives as texture representation. Since the dependency of primitives reveals the inherent structural properties of texture, the generated texture representation shows good robustness to spatial deformations and brightness change. Extensive experiment results suggest that our proposed DSR-Net approach is not only an effective solution for texture recognition, but also shows great potential in many visual applications. Incorporating with visual texture attributes perceiving method, such as [31], will be our future work.

Acknowledgment

This work was supported by the National Key R&D Program of China under Grant 2017YFB1300201, the National Natural Science Foundation of China (NSFC) under Grants 61872327, 61622211, U19B2038 and 61620106009 as well as the Fundamental Research Funds for the Central Universities under Grant WK2380000001 and WK2100100030.

References

[1] Barbara Caputo, Eric Hayman, and P Mallikarjuna. Class-specific material categorisation. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1597–1604, 2005.

[2] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3606–3613, 2014.

[3] Mircea Cimpoi, Subhransu Maji, and Andrea Vedaldi. Deep filter banks for texture recognition and segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3828–3836, 2015.

[4] Gabriella Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV*, pages 1–2, 2004.

[5] Xiyang Dai, Joe Yue-Hei Ng, and Larry S Davis. Fason: First and second order information fusion network for texture recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7352–7360, 2017.

[6] Dalal, Navneet, Triggs, and Bill. Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 886–893, 2005.

[7] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015.

[8] Robert M Haralick, Karthikeyan Shanmugam, et al. Textural features for image classification. *IEEE Transactions on systems, man, and cybernetics*, pages 610–621, 1973.

[9] Han Hu, Zheng Zhang, Zhenda Xie, and Stephen Lin. Local relation networks for image recognition. *arXiv preprint arXiv:1904.11491*, 2019.

[10] Hervé Jégou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. Aggregating local descriptors into a compact image representation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3304–3311, 2010.

[11] Bela Julesz. Textons, the elements of texture perception, and their interactions. *Nature*, 290(5802):91, 1981.

[12] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 554–561, 2013.

[13] Tsung-Yu Lin and Subhransu Maji. Visualizing and understanding deep texture representations. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2791–2799, 2016.

[14] Li Liu, Jie Chen, Paul Fieguth, Guoying Zhao, Rama Chellappa, and Matti Pietikainen. From bow to cnn: Two decades of texture representation for texture classification. *International Journal of Computer Vision (IJCV)*, pages 1–36, 2018.

[15] David G Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, pages 91–110, 2004.

[16] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research (JMLR)*, pages 2579–2605, 2008.

[17] P Mallikarjuna, Alireza Tavakoli Targhi, Mario Fritz, Eric Hayman, Barbara Caputo, and Jan-Olof Eklundh. The kth-tips2 database. *Computational Vision and Active Perception Laboratory (CVAP), Stockholm, Sweden*, 2006.

[18] Timo Ojala, Pietik, Matti Inen, and Topi. *Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns*. Springer Berlin Heidelberg, 2000.

[19] Florent Perronnin, Jorge Sánchez, and Thomas Mensink. Improving the fisher kernel for large-scale image classification. In *European Conference on Computer Vision (ECCV)*, pages 143–156, 2010.

[20] Tingting Qiao, Jing Zhang, Duanqing Xu, and Dacheng Tao. Mirrorgan: Learning text-to-image generation by redescription. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1505–1514, 2019.

[21] Yong Rui, Thomas S Huang, and Shih-Fu Chang. Image retrieval: Current techniques, promising directions, and open issues. *Journal of Visual Communication and Image Representation (JVCIR)*, pages 39–62, 1999.

[22] Lavanya Sharan, Ce Liu, Ruth Rosenholtz, and Edward H Adelson. Recognizing materials using perceptually inspired features. *International Journal of Computer Vision (IJCV)*, pages 348–371, 2013.

[23] Lavanya Sharan, Ruth Rosenholtz, and Edward Adelson. Material perception: What can you see in a brief glance? *Journal of Vision*, pages 784–784, 2009.

[24] Yang Song, Fan Zhang, Qing Li, Heng Huang, Lauren J O'Donnell, and Weidong Cai. Locally-transferred fisher vectors for texture classification. In *IEEE International Conference on Computer Vision (ICCV)*, pages 4922–4930, 2017.

[25] Manik Varma and Andrew Zisserman. Texture classification: Are filter banks necessary? In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages II–691, 2003.

[26] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.

[27] Jia Xue, Hang Zhang, and Kristin Dana. Deep texture manifold for ground terrain recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 558–567, 2018.

[28] Jia Xue, Hang Zhang, Kristin J Dana, and Ko Nishino. Differential angular imaging for material recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6940–6949, 2017.

[29] Ze Yang, Tiange Luo, Dong Wang, Zhiqiang Hu, Jun Gao, and Liwei Wang. Learning to navigate for fine-grained classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 420–435, 2018.

[30] Zheng-Jun Zha, Xian-Sheng Hua, Tao Mei, Jingdong Wang, Guo-Jun Qi, and Zengfu Wang. Joint multi-label multi-instance learning for image classification. In *2008 ieee conference on computer vision and pattern recognition*, pages 1–8. IEEE, 2008.

[31] Wei Zhai, Cao Yang, Zhang Jing, and Zheng-Jun Zha. Deep multiple-attribute-perceived network for real-world texture recognition. In *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[32] Hang Zhang, Kristin Dana, Jianping Shi, Zhongyue Zhang, Xiaogang Wang, Ambrish Tyagi, and Amit Agrawal. Context encoding for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7151–7160, 2018.

[33] Hang Zhang, Jia Xue, and Kristin Dana. Deep ten: Texture encoding network. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 708–717, 2017.

[34] Hanwang Zhang, Zheng-Jun Zha, Yang Yang, Shuicheng Yan, and Tat-Seng Chua. Robust (semi) nonnegative graph embedding. *IEEE transactions on image processing*, 23(7):2996–3012, 2014.

[35] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2881–2890, 2017.

[36] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 633–641, 2017.