Enhancing Texture Representation with Deep Tracing Pattern Encoding

Zhile Chen[a,c], Yuhui Quan[a,c,*], Ruotao Xu[a,c,**], Lianwen Jin[b], Yong Xu[a,d,e]

[a] *School of Computer Science and Engineering, South China University of Technology, Guangzhou, China*
[b] *School of Electronic and Information Engineering, South China University of Technology, Guangzhou, China*
[c] *Pazhou Laboratory, Guangzhou, China*
[d] *Peng Cheng Laboratory, Shenzhen, China*
[e] *Communication and Computer Network Laboratory of Guangdong, Guangzhou, China*

Abstract

Texture representation is a challenging problem due to the complex underlying physics of texture as well as the variations caused by changes in viewpoint. Recent progress in texture analysis has been made by the power of convolutional neural networks (CNNs) in feature learning. However, most current methods aggregate the features from the last convolutional layer of the CNN to obtain a global feature vector, which fails to leverage shallow low-level visual cues and cross-layer feature patterns, limiting their performance. In this paper, we propose to trace the features generated along the convolutional layers via a histogram of local 3D invariant binary patterns, called deep tracing patterns. This leads to a highly discriminative yet robust global feature representation module. Building such a module into a CNN backbone, we develop an effective approach for texture recognition. Extensive experiments on six benchmark datasets show that the proposed approach provides a discriminative and robust texture descriptor, with state-of-the-art performance achieved.

Keywords: Texture recognition, Deep learning, Feature encoding

1. Introduction

Texture recognition is a fundamental task in computer vision, with a wide range of applications such as material classification [1], terrain recognition [2], face analysis [3], and object recognition [4]. One of the main challenges of texture recognition lies in its high intra-class variations, which originates from its complex underlying physics and the various spatial transforms caused by viewpoint changes. Moreover, the diverse and even contradictory nature of texture, such as uniformity vs. deformability, regularity

*Zhile Chen and Yuhui Quan contribute equally.
**Corresponding author
Email addresses: cszhilechen@mail.scut.edu.cn (Zhile Chen), csyhquan@scut.edu.cn (Yuhui Quan), xrt@scut.edu.cn (Ruotao Xu), eelwjin@scut.edu.cn (Lianwen Jin), yxu@scut.edu.cn (Yong Xu)

vs. randomness, and fine-scale patterns vs. coarse-scale patterns, also pose a significant difficulty to the feature representation of texture. An ideal texture representation should possess both high discriminability for capturing the underlying physics of textures and high robustness to different types of spatial transforms and disturbances.

Texture recognition has been extensively investigated in the past decades. Traditional methods typically extract local features (*e.g.*, using SIFT [5] and LBP [6]) from a texture image, and then aggregate them into a global representation (*e.g.*, using BoW [7] and fractal statistics [8]) for classification. Recently, with the advent of convolutional neural networks (CNNs) and their success in universal image classification, numerous CNN-based texture recognition approaches have emerged. However, texture image recognition presents unique challenges that may not be adequately addressed by generic CNNs, resulting in suboptimal performance. Consequently, to boost the CNNs' performance in texture recognition, specialized modules are developed to bridge the gap between the general CNN architecture and the specific demands of texture recognition.

For instance, many existing methods address the tension between the need for robustness against spatial transformations in global aggregation and the spatially-indexed fully-connected (FC) layers. To achieve an orderless representation, global average pooling (GAP) [9, 10] is placed on the top-layer features to ensure invariance to spatial deformations. Nevertheless, the straightforward use of GAP may yield less descriptive features, raising the question of how to maintain discriminative power during aggregation. To address this problem, several aggregation techniques based on specific statistics have been proposed, such as FV-CNN [11], DeepTEN [12], and many others.

While numerous endeavors have been made to enhance global aggregation for texture classification beyond generic CNNs, local feature extraction, another crucial step in texture recognition, has received comparatively less attention during the CNN era. In this work, we investigate the exclusive requirements of local texture description against the general CNN design. Towards this end, we propose a so-called Deep Tracing Pattern (DTP) module that aims at extracting robust and discriminative local descriptors for texture recognition. The main idea of the proposed DTP module is illustrated in Fig. 1. Integrating the DTP module into a ResNet backbone, we develop DTPNet, an effective deep model for texture recognition, which exhibits state-of-the-art performance in the experiments.

1.1. Motivations and Main Idea

In this work, we explore the distinctive characteristics of texture recognition, which should be considered in a local feature extraction scheme but ignored by a generally-designed CNN. A local textural feature extraction module is then developed to fit these properties and boost the texture recognition performance. The following characteristics are taken into account:

Fine-grained details are crucial for texture recognition. In contrast to object or scene recognition where coarse-scale structures play a dominant role, fine-scale details typically can be decisive clues for texture recognition, as texture usually relates to the "micro-structures" of an image. However, most existing methods concentrate on the manipulation of the top-layer features, which inevitably overlooks some textural

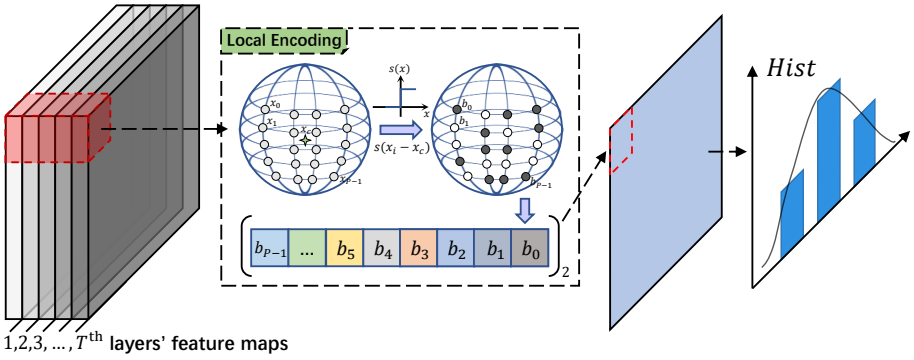


Figure 1: Illustration of the basic idea of deep tracing pattern (DTP), which takes feature maps from multiple layers as input, encodes each local patch with binary codes, and aggregates them into a histogram-based global feature.

characteristics in the fine scales due to the involved downsampled operations or learned low-pass convolutional kernels. Hence, we exploit the local features on multiple layers, taking both the shallow and the deep features into consideration.

Evolution rules across layers/scales provide useful clues for texture recognition. As shown in [13], the evolution rule of texture structure across scales is useful for recognition. However, these clues are seldom explicitly modeled for local feature extraction in most existing CNNs. A CNN builds up a hierarchical representation of an image based on a series of convolutional layers. The feature maps from one layer to the next encode texture structures from a smaller to a larger scale. That is, we can treat the generation of feature maps of a texture image along CNN layers as a dynamic evolution process in scales, which should provide useful clues for texture recognition. Fig. 2 illustrates this concept, where the cross-layer features are obtained by stacking the size-normalized feature maps from different layers and the feature vectors are extracted along the stacked dimension for analysis. As plotted in the bottom-left part, the vectors of the stone brick patches exhibit similar trends along the layers, while those of stone brick and grass differ noticeably. The bottom-right plot visualizes these feature vectors via t-SNE [14], where the features of the same classes are clustered, while those of different classes are separated. This observation motivates us to trace the patterns along CNN layers and explore them for more discriminative local features.

Texture recognition has a high demand for robustness against rotation and illumination changes. Real-world textures can occur at arbitrary rotations and they may be subjected to varying illumination conditions, highlight the importance of robustness for a texture descriptor. However, in conventional CNN pipelines, no explicit constraints regarding such robustness are imposed. As a result, the robustness of the local features is hard to guarantee when facing various transforms. This necessitates a robust local encoding scheme, which can guarantee the robustness of the local features.

Motivated by the aforementioned observations, our proposed DTP module traces the well-learned CNN features across layers and encodes them with an LBP-inspired

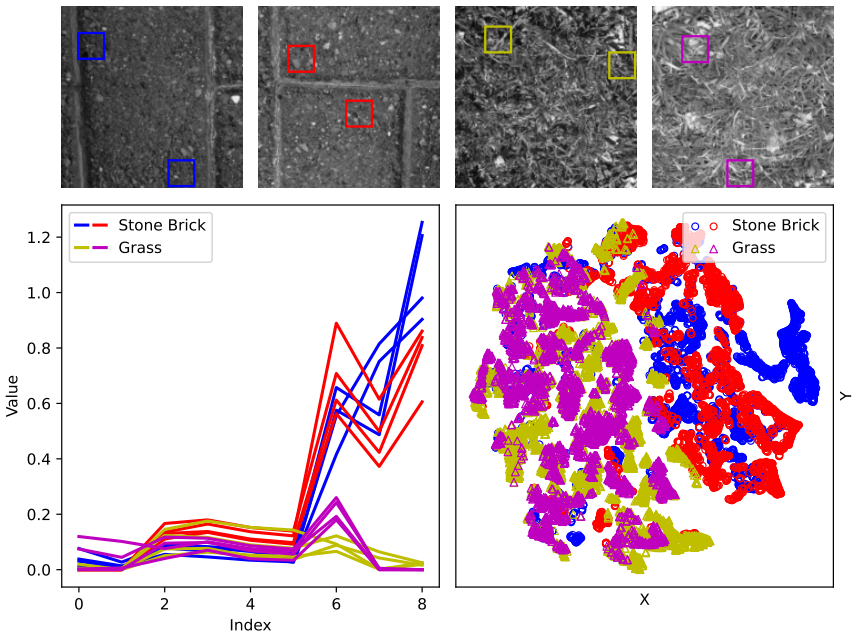


Figure 2: Illustration of the evolutionary patterns across layers. Top: Four textured images from two classes "Stone Brick" and "Grass"; Bottom left: Plot of the cross-layer feature vectors, where the colors denote the corresponding sources patches marked in the top row; Bottom right: t-SNE visualization of the cross-layer features of the images using ℓ_2 distances. All the features used are extracted from ResNet50.

scheme. The fundamental idea is illustrated in Fig. 1, where features predicted by different layers, from shallow to deep, are collected, resized, and concatenated to form grouped cross-layer feature tensors. Then, the local patches in the grouped tensors are encoded into 3D binary codes to capture both the spatial-local and the cross-layer textural characteristics, followed by a spatial pyramid histogram for multi-scale global feature extraction. Benefiting from such a pipeline, the resulting DTPNet that integrates DTP into a ResNet backbone is effective for texture recognition.

1.2. Contributions

In summary, our contributions are as follows.

Exploiting cross-layer features in local feature extraction. Unlike most existing approaches *e.g.* [12, 15, 16], which primarily focus on improving the global aggregation scheme, this work investigates local feature extraction by exploiting the cross-layer features in CNNs. There are some approaches (*e.g.* [17]) that merge feature maps from different layers into a new one, on which global feature aggregation is applied. In comparison, the proposed DTP module extracts the local pattern in the cross-layer statistics, including the spatial details and the trends along layers. Our work thus can inspire further studies on the evolution of CNN features for other computer vision tasks.

Introducing robustness for local feature extraction. Texture features need to be robust against deformation and disturbances as real-world textures can occur in various rotation or illumination situations. However, the robustness of local features is seldom addressed in the CNN period. In this paper, we construct an LBP-inspired local encoding scheme for feature extraction, which explicitly enforces rotation invariance and gray-scale robustness to extracted local codes.

State-of-the-art performance achieved. Incorporating the DTP module into CNN backbones for boosting the robustness and discriminability of texture representation, the proposed DTPNet achieves state-of-the-art results on several benchmark datasets in classification and retrieval tasks.

2. Related Works

As a fundamental computer vision topic, texture recognition has been researched for years. Interested readers are referred to [18] for a comprehensive survey. For a concentrated introduction, we review the works related to ours in this section.

2.1. Handcrafted Features for Texture Recognition

Early approaches usually adopt handcrafted designs for texture recognition. Histogram of textons (*e.g.* [19, 20]) and BoVW of texture (*e.g.* [7]) are two representative frameworks of this route. Both frameworks include two critical steps: local patch encoding and global feature aggregation. For global feature aggregation, the researchers make efforts on introducing additional discriminativeness beyond the naive histograms. The VLAD [21] aggregates the first-order statistics on the accumulated differences between a local descriptor and its correspondences. The Fisher vector [11] further introduces the 2nd-order statistics for encoding. The MFS [8] employs fractal-geometry-based statistics for global feature aggregation. Regarding local encoding, on the contrary, many researchers aim at enhancing the invariance or robustness of the local descriptors. Lowe *et al.* [5] introduce the scale-invariant feature transform(SIFT) by locating the local minimum/maximum in the scale pyramid. Lazebnik *et al.* [7] propose the rotation-invariant feature transform(RIFT), which maintains rotation invariance via the histograms of relative gradient orientation in rings. Ojala *et al.* [6] introduce the local binary pattern, which achieves gray-scale invariance by thresholding a local neighborhood at the gray value of the center pixel into a binary pattern. Rotational invariance is further obtained by maximal circular bit-shift codes. Inspired by LBP, we also introduce the binarization scheme and circular-shift maximization mechanism for describing the cross-layer feature maps in CNN.

2.2. CNN-based Texture Recognition

Inspired by the success of CNN in the computer vision realm, a growing number of methods utilize CNN as a powerful feature extractor for texture recognition. One of the seminal works can be traced back to the work of Bruna *et al.* [22], which implements scattering transformation with convolutional layers for texture classification. Though relishing the invariance brought by scattering transform to certain deformations, their CNN is not learned but with predetermined weights, *e.g.* basic wavelet filters, and thus

cannot leverage the power of deep learning. To introduce the learning power of CNNs, Cimpoi *et al.* [23] use a plain CNN architecture followed by fully connected (FC) layers for texture recognition. However, as demonstrated in [9], a CNN with FC layers is not a good choice for texture recognition. A probable reason is that texture recognition has a higher demand for robustness to spatial transformation, which is against the spatial-indexing nature of FC layers. To improve the robustness, Andrearczyk *et al.* [9] and Fujieda [10] apply global average pooling(GAP) to achieve invariance to the spatial arrangement. However, simple spatial accumulation may lead to low discriminativeness. To address this issue, Lin and Maji [24] apply bi-linear pooling which exploits the correlation between channels. Inspired by the handcrafted features aggregation mechanisms, Perronnin *et al.* [11] introduce the FV to encode the convolutional features from a pre-trained CNN. Owing to the non-differentiable nature of FV, the weights of the pre-trained backbone CNN in [11] cannot be fine-tuned. To enable end-to-end learning, Zhang *et al.* [12] generalize VLAD and FV, and integrates an encoding layer on top of convolutional layers into CNN. Xue *et al.* [2] combine DeepTEN and GAP, by which local appearance and global context are simultaneously captured. The combination is done by applying bi-linear pooling to the features pooled from DeepTEN and GAP. Bu *et al.* [25] propose a locality-aware coding layer that performs dictionary learning and feature encoding on convolutional features. Note that the methods mentioned above utilize orderless pooling for aggregation, which may ignore the spatial layout of local features. To encode the spatial dependency between local primitives, Zhai *et al.* [26] propose a model called DSRNet with a dependency learning module, which exploits the spatial dependency among texture primitives for capturing structural relations between local features. Based on multi-fractal geometry, Xu *et al.* [15] leverage the hierarchical fractal analysis to encode the fractal property of spatial arrangement within the CNN's feature maps. Mao *et al.* [16] propose a deep residual pooling network that combines a residual encoding module that preserves spatial information and an aggregation module that generates orderless features.

Enhancing the discriminativeness while ensuring robustness has always been the objective for texture representation. The methods mentioned above all achieved the improvement with better manipulation of the top-layer feature maps at a coarse scale. However, different from object or scene recognition, fine-scale details are also important for texture recognition. To encode feature maps in different scales, Hu *et al.* [27] propose to encode multiple-layer features, which performs feature aggregation on different convolutional blocks individually and fuse the results by an FC layer, Chen *et al.* [17] also consider feature maps across layers, which learns an implicit model of statistical self-similarity carried within the CNN's feature maps.

The mentioned methods all make their efforts on improving the global feature aggregation, while the counterpart, local encoding gains little attention in the CNN period. In this work, we make our efforts on local feature encoding by introducing the local binary pattern on the cross-layer features.

2.3. LBP and LBP-inspired CNNs

Local binary pattern (LBP) is originally proposed by Ojala *et al.* [6], serving as a significant and typical tool for image classification, e.g. texture recognition [28] and face recognition [29]. The key idea of LBP is to extract the feature from the

differential distribution of a texture image based on texture spectrum modeling. For addressing the problem caused by scale transformation and noise interference, Luo *et al.* [28] propose an LBP-based texture descriptor named SNELBP which combines the advantages of multi-scale Gaussian filter, MRELBP [30], noise-robust and scale-invariant histogram. Shu *et al.* [31] construct an LBP-based texture descriptor called GRLBP to analyze the nature of pixel intensity distribution in local neighborhoods with the assistance of global information for higher discriminativeness. With the rapid development of CNNs, a number of approaches focus on the combination between the handcrafted features (*e.g.* LBP and HOG) and the CNN features as a discriminative texture representation. Inspired by LBPs, Xu *et al.*[32] propose the so-called local binary convolution (LBC), which exploits a set of fixed pre-defined binary differential filters for feature extraction. Though the LBC can afford significant parameter saving, rotation invariance is not considered and introduced in LBC. Aiming at rotation invariance, Zhang *et al.* [33] propose a deep architecture named Local Binary orientation Module (LBoM), where the orientation feature is learned and then discarded for robust features. Unlike the fusion schemes mentioned above, our proposed method is the pathfinder for adopting the LBP descriptor on the cross-layer features from CNN.

3. Proposed Method

3.1. Overview

In this paper, we proposed a novel Deep Tracing Pattern Network (DTPNet) for real-world texture recognition, which captures discriminative and robust local features by tracing the local patterns across different network layers. The overall architecture of DTPNet is illustrated in Fig. 3. The proposed network is built upon the ResNet backbones [34], which consists of a series of residual blocks(RBs), followed by an average pooling layer to extract the global feature, and a fully-connected layer with Softmax for label prediction. To capture the texture patterns across layers, we proposed a deep tracing pattern (DTP) module, which takes the features from T selected residual blocks (RBs) in the backbone as inputs and outputs a DTP feature. Formally, the DTP module can be written as

$$\boldsymbol{f}_{\text{DTP}} = \text{DTP}(\mathcal{F}_1, \cdots, \mathcal{F}_T), \tag{1}$$

where $\mathcal{F}_i \in \mathbb{R}^{H_i \times W_i \times D_i}$ denote the feature tensor output by the i-th selected RB, and $\boldsymbol{f}_{\text{DTP}}$ is outputted global DTP feature. The DTP feature $\boldsymbol{f}_{\text{DTP}}$ is then concatenated with the GAP's output of the backbone for classification. The details of the proposed DTP module are given as follows.

3.2. Deep Tracing Pattern Module

The DTP module consists of three phases: cross-layer feature grouping, robust local feature encoding and spatial pyramid histogram calculation.

Cross-layer feature grouping Recall that the spatial size $H_i \times W_i$ and the channel number D_i vary across different RBs. In the first phase, the DTP module firstly normalizes the inputs $\mathcal{F}_i \in \mathbb{R}^{H_i \times W_i \times D_i}$ into the same shape. Toward this end, we apply

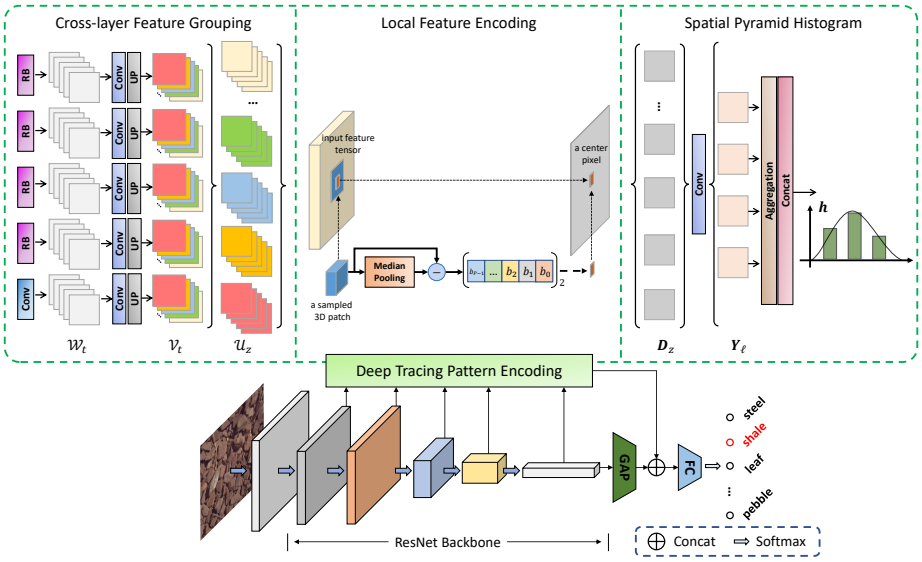


Figure 3: The overall architecture of DTPNet. The proposed DTP module is integrated into a ResNet backbone, which groups the feature maps from multiple layers through the cross-layer feature grouping scheme (left), extracts local features using the DTP encoding (middle), and aggregates the global feature using soft spatial pyramid histogram (right).

a 1×1 convolution to transform each $\mathcal{F}_i$ into a D-channel one and then upsample it into the same spatial resolution $H \times W$, with $H = \max_i H_i$ and $W = \max_i W_i$. In the implementation, we utilize bilinear interpolation for upsampling.

Let $\mathcal{N}_1, \cdots, \mathcal{N}_T \in \mathbb{R}^{H \times W \times D}$ denote the size-normalized feature tensors. To capute the patterns along layers, We reorganize $\mathcal{N}_i$s into D feature tensors as

$$\mathcal{G}_d = [\boldsymbol{N}_1^d; \boldsymbol{N}_2^d; \cdots; \boldsymbol{N}_T^d] \in \mathbb{R}^{H \times W \times T}, \ \forall d, \tag{2}$$

where $\boldsymbol{N}_i^d$ denotes the d-th channel of $\mathcal{N}_i$ and $[\cdot;\cdot]$ denotes the concatenation along the channel dimension. In other words, the feature maps from different layers but the same channels are grouped into tensors, so the patterns along layers can be extracted from the reorganized feature tensor $\mathcal{G}_d$s.

Robust local feature encoding Each local neighborhood located in $\mathcal{G}_d$ contains rich structures that encode complex spatial characteristics and along-layer patterns of textures. The second phase of the DTP module aims at extracting discriminative and robust features for each local neighborhood. Borrowing the idea of LBPs which has shown their effectiveness in characterizing local patterns, we sample 3D local patches in each position of each tensor $\mathcal{G}_d$ with a sliding window, and extract a binary code for each 3D patch. Specifically, for some sampled patch $\mathcal{X} \in \mathbb{R}^{k \times k \times T}$, a set of equally spaced pixels on a ball around the patch center with radius r is sampled. Let $\boldsymbol{x} = [x_1, x_2, \cdots, x_P]$ denotes the values of sampled pixels and M denotes the median value

of the patch. The binary code of the patch $\mathcal{X}$ is then calculated as

$$\phi(\boldsymbol{x}, M) = \sum_{p=1}^{P} s(x_p - M)2^{p-1}. \tag{3}$$

Here, the median value is used for its higher robustness to noises, and the differences between each pixel and median value encourage robustness against the illumination changes. For the robustness against rotation, we further calculate minimized code among different permutations as

$$\psi(\mathcal{X}) = \min_{\pi \in \Pi} \phi(\pi \circ \boldsymbol{x}, M), \tag{4}$$

where Π is a set of permutation operations that represents rotations on the sampling ball. The minimization among these permutations leads to rotation-invariance on the final binary codes.

For local feature encoding, we calculate the binary codes $c_{p,t} = \psi(\mathcal{X}_{p,t})$ for each patch $\mathcal{X}_{p,t}$ that locates at the p-th position of the t-th grouped tensor $\mathcal{G}_t$. Then, the binary codes for the same position but different tensors are concatenated into feature vectors $\boldsymbol{c}_p = [c_{p,1}, \cdots, c_{p,T}] \in \mathbb{R}^T$. A linear projection is further applied to each feature vector, which refines and transforms them into lower dimensions. Formally, the calculation of the local features can be written as

$$\boldsymbol{f}_p = \boldsymbol{W}\boldsymbol{c}_p, \forall p \in \mathbb{I}, \tag{5}$$

where $\mathbb{I} = \{1, \cdots, H\} \times \{1, \cdots, W\}$ denotes the spatial indices and $\boldsymbol{W} \in \mathbb{R}^{N \times T}$ denotes the projection matrix. In our implementation, the linear projection is implemented with 1×1 convolutions, and the feature length N is set to be $4, 16$ for the ResNet18 and ResNet50 backbones, respectively.

Spatial pyramid histogram In the former phase, a set of local features $\{\boldsymbol{f}_p | \, p \in \mathbb{I}\}$ is produced. The global feature aggregation phase aims at aggregating these local features into a single global feature vector. Instead of the simple global average pooling, we calculate a soft histogram for each dimension in feature $\boldsymbol{f}_p$s. Specifically, for the i-th dimension, the global histogram can be calculated as $\boldsymbol{h}_i = \text{Hist}(\{\boldsymbol{f}_p | p \in \mathbb{I}\})$, whose k-th element is defined as

$$h_{i,k} = \sum_{p \in \mathbb{I}} \frac{\exp(-s_{i,k}^2 \cdot (f_{p,k} - \beta_{i,k})^2)}{\sum_{k'=1}^{K} \exp(-s_{i,k'}^2 \cdot (f_{p,i} - \beta_{i,k'})^2)}, \tag{6}$$

for $k = 1, \cdots, K$, where $f_{p,i}$ denotes the ith element of $\boldsymbol{f}_p$, $\beta_{i,k}$s are learnable bin centers , and $s_{i,k}$s are the learnable scaling factors. Note that in (6), the global histogram completely discards the spatial orders, which may limit the descriptive capacity of the representation. To overcome the problem, we apply the idea of the spatial pyramid in the histogram calculation. Specifically, we partition the image into $2^\ell \times 2^\ell$ segments for the ℓ-th scale, with $\ell = 0, 1, \cdots, L$, resulting in $M = (4^{L+1} - 1)/3$ segments. The soft histograms are then calculated for all the segments and finally concatenated into a vector representation of the whole image. Let $\mathbb{I}^\ell$ denotes the indices of the ℓ segment, and

$\boldsymbol{h}^{(\ell)} = \text{Hist}(\{\boldsymbol{f}_p\}|p \in \mathbb{I}^\ell\})$ denotes the corresponding histogram. The global feature for the i-th channel is then constructed by concatenating these histograms as

$$\boldsymbol{f}_i = [\boldsymbol{h}_i^{(0)}, \cdots, \boldsymbol{h}_i^{(M-1)}].$$

The bin centers and scaling factors in $\boldsymbol{h}_i^{(\ell)}$ are shared for different ℓ but vary across different i. The final DTP feature vector is then defined as $\boldsymbol{f}_{\text{DTP}} = [\boldsymbol{f}_1, \cdots, \boldsymbol{f}_L]$, which is then concatenated with the GAP's output for classification.

Remark 1. *Multi-scale pooling has also been considered in general image classification, which generates multi-scale patches and concatenates the descriptors from different scales for classification. In comparison, the proposed DTP module does not consider the multi-scale representation of CNN features in each layer but considers the dynamics of the evolution flow of feature maps along the CNN depth dimension. Beyond the multi-resolution analysis, DTP also considers the flow dynamics along layers, which provides implicit discriminative clues for texture recognition. In addition, for robustness to image transforms like rotation, we adopt the idea from the LBP-inspired module to obtain a rotation-robust descriptor to characterize those dynamics.*

3.3. Implementation Details

In the implementation of our DTPNet, the parameters during the cross-layer feature grouping are set as: $T = 5$, $D = 16$. While in the local feature encoding phase, the parameters are set as: $k = 5$, $P = 14$, $r = 2$; For the spatial pyramid histogram, the paramters are set as: $L = 2$, $K = 8$.

Following existing works, ResNet18 and ResNet50 [34] are used as the backbones. Our model is trained using an SGD optimizer with a momentum of 0.9 with batch size set to 32. The learning rate is initialized to 1×10^{-3} and decreased in the cosine decay scheme. The model is trained for 30 epochs for convergence. The ResNet backbone is initialized with pre-trained models and other network parameters are initialized by the default Kaiming [35]. The model is then trained with 30 epochs for convergence. Following [12, 2], we resize the images to 256×256 and then crop them to 224×224 during training and test. All training images are randomly horizontally flipped with a probability of 0.5 for data augmentation. We implement our model with PyTorch and conduct the following experiments on a single RTX Titan GPU.

4. Experiments

4.1. Datasets and Protocols

Benchmark datasets We evaluate our model on six benchmark datasets, whose details are as follows. *(a)* Ground Terrain in Outdoor Scenes (GTOS) [36] is a dataset of outdoor ground materials with 40 categories, with a training/testing split given. *(b)* GTOS-Mobile [2] is a dataset collected from GTOS via mobile phone, which consists of 100011 material samples from 31 categories. *(c)* Materials in Context 2500 (MINC-2500) [1] is a dataset of 23 material categories, each of which contains 2500 images. It provides five training/test splits. *(d)* KTH-TIPS2b [19] is a dataset composed of 4752 images from 11 material categories. *(e)* Describable Texture Dataset

(DTD) [37] contains 47 categories of wild textures, with 120 images per category. It provides 10 preset splits into equally-size training, validation, and test sets. *(f)* Flickr Material Dataset (FMD) [38] is composed of 10 different material categories, with 100 images each category. These datasets cover a broad spectrum of textures. See Fig. 4 for some examples.

Figure 4: Samples images from six datasets.

Protocols Following the recent works [26, 17], we use the pre-set splits on GTOS and MINC-2500, and the splits as [17] for DTD. As for KTH-TIPS2b and FMD, each dataset is randomly divided into 10 splits with recommended split size. The mean results across splits are reported. In classification experiments, the results over multiple runs are collected, and their averages and standard deviations (std) are recorded in the form of "mean $\pm$ std.%". As for the compared methods, the results are directly quoted from the recent works (mainly from [17]), otherwise, blanks are left.

4.2. Comparison against State-Of-The-Arts

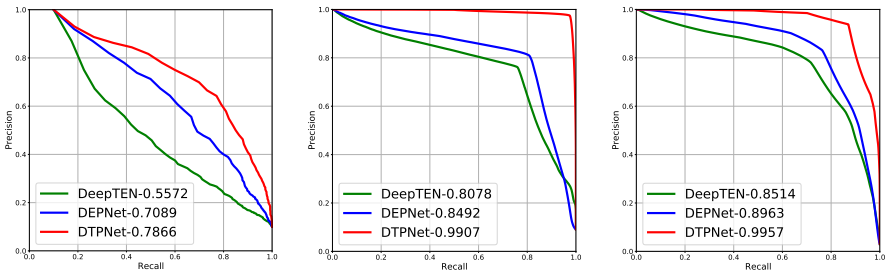
Results on texture recognition We compare the performance of our method with 11 SOTA CNN-based texture recognition approaches, including DeepTEN [12], DEPNet [2], LSCNet [25], MAPNet [39], DSRNet [26], HistNet [40], CLASSNet [17], FENet [15], RPNet [16], TEMNet [41], and MSBFEN [42]. Table 1 summarizes the classification results on six benchmark datasets. When incorporated with the ResNet18 backbone, the proposed DTPNet obtains the best results among four of six datasets. The proposed method outperforms the second-best performer with significant improvements of 2.6% accuracy on FMD and 3.4% accuracy on GTOS-mobile. With ResNet50 as the backbone, our DTPNet ranks first on half of the datasets including KTH, FMD, and GTOS-Mobile. The overall results demonstrate the effectiveness of the DTP encoding.

Results on texture retrieval We also evaluate the proposed method in the texture retrieval task and compare it with features produced by DEPNet [2] and DeepTEN [12]. For evaluation, each query image is fed into the proposed DTPNet and the concatenated feature vector before the FC layer is used for feature matching in terms of cosine

Table 1: Performance comparison of different methods in terms of classification accuracy (%). The result with the highest average accuracy on each dataset is marked in **bold**.

Method	Backbone	DTD	KTH	FMD	MINC	GTOS	GTOS-Mobile
DeepTEN [12]	ResNet18	-	-	-	-	-	76.1
DEPNet [2]		-	-	-	-	-	82.2
LSCNet [25]		-	-	76.3	-	-	-
MAPNet [39]		69.5 ± 0.8	80.9 ± 1.8	80.8 ± 1.0	-	80.3 ± 2.6	83.0 ± 1.6
DSRNet [26]		71.2 ± 0.7	81.8 ± 1.6	81.3 ± 0.8	-	81.0 ± 2.1	83.7 ± 1.5
HistNet [40]		-	-	-	-	-	79.8 ± 0.8
CLASSNet [17]		71.5 ± 0.4	85.4 ± 1.1	82.5 ± 0.7	80.5 ± 0.6	84.3 ± 2.2	85.3 ± 1.3
FENet [15]		69.6 ± 0.1	86.6 ± 0.1	82.3 ± 0.3	80.6 ± 0.1	83.1 ± 0.2	85.1 ± 0.4
RPNet [16]		71.6 ± 0.7	86.7 ± 2.7	83.3 ± 3.0	79.0 ± 0.5	83.3 ± 2.2	76.6 ± 1.5
TEMNet [41]		70.2 ± 1.0	$\mathbf{88.0 \pm 3.5}$	83.1 ± 0.8	$\mathbf{80.8 \pm 0.4}$	83.4 ± 1.9	83.6 ± 0.0
DTPNet		$\mathbf{71.8 \pm 0.7}$	86.7 ± 1.3	$\mathbf{85.7 \pm 0.9}$	80.7 ± 0.4	$\mathbf{84.8 \pm 2.4}$	$\mathbf{87.0 \pm 1.2}$
DeepTEN [12]	ResNet50	69.6	82.0 ± 3.3	80.2 ± 0.9	81.3	84.5 ± 2.9	-
DEPNet [2]		73.2	-	-	82.0	-	-
LSCNet [25]		-	-	81.2	-	-	-
MAPNet [39]		76.1 ± 0.6	84.5 ± 1.3	85.2 ± 0.7	-	84.7 ± 2.2	86.6 ± 1.5
DSRNet [26]		77.6 ± 0.6	85.9 ± 1.3	86.0 ± 0.8	-	85.3 ± 2.0	87.0 ± 1.5
HistNet [40]		72.0 ± 1.2	-	-	82.4 ± 0.3	-	-
CLASSNet [17]		74.0 ± 0.5	87.7 ± 1.3	86.2 ± 0.9	84.0 ± 0.6	85.6 ± 2.2	85.7 ± 1.4
FENet [15]		74.2 ± 0.1	88.2 ± 0.2	86.7 ± 0.2	84.0 ± 0.1	85.7 ± 0.1	85.2 ± 0.4
RPNet [16]		73.0 ± 0.6	87.2 ± 1.8	87.2 ± 2.4	81.6 ± 0.4	83.6 ± 2.3	77.9 ± 0.3
MSBFEN [42]		$\mathbf{77.8 \pm 0.5}$	86.2 ± 1.1	86.4 ± 0.7	$\mathbf{85.3 \pm 0.7}$	$\mathbf{86.4 \pm 1.8}$	87.6 ± 1.6
DTPNet		73.5 ± 0.4	$\mathbf{88.5 \pm 1.6}$	$\mathbf{87.8 \pm 1.3}$	83.7 ± 0.3	86.1 ± 2.5	$\mathbf{88.0 \pm 1.2}$

similarity. The average precision and recall over all the query images are collected. See Fig. 5 for the precision-to-recall (PR) curve by varying S from 1 to the number of samples as well as the corresponding mean average precision (mAP). The figure shows that the retrieval result using the deep representation from DTPNet significantly outperforms the ones from DeepTEN and DEPNet, indicating the excellent discriminativeness and robustness of the feature learned by the DTP module. See also the reported mAP in the plots, where the proposed DTPNet also achieves obvious improvements over the competitors, *i.e.*more than 0.08mAP on FMD, 0.15 mAP on KTH and 0.10 on GTOS.



(a) FMD (b) KTH (c) GTOS

Figure 5: PR curve of the retrieval results on 3 datasets. The mAPs are also reported in the legends. The ResNet18 backbone is used.

Table 2: Complexity, training and test time comparison of different models.

Model	#Params (≈, M)		FLOPs (≈, G)		Training time (min.)		Test time (10^{-3} sec.)	
	ResNet18	ResNet50	ResNet18	ResNet50	ResNet18	ResNet50	ResNet18	ResNet50
Backbone	11.19	23.57	1.82	4.11	2.52	6.83	0.63	1.27
DeepTEN	11.37	23.90	1.82	4.12	2.57	6.88	0.60	1.29
DEPNet	12.01	25.56	1.82	4.11	2.63	6.98	0.63	1.30
CLASSNet	11.23	23.70	1.83	4.14	22.36	26.87	7.40	8.22
FENet	11.51	23.93	1.82	4.12	3.21	7.97	0.73	1.46
DTPNet	11.23	23.70	1.83	4.14	13.12	21.74	3.70	5.65

Model complexity comparison For evaluation of complexity, we compare our method with FENet, CLASSNet, DeepTEN, DEPNet and ResNet backbone in terms of *(a)* model size measured by the number of model parameters (#Params); and *(b)* computational complexity measured by the Floating-Point Operations per second(FLOPs) of the model. See Table 2 for the comparison results. The results show that our model contains the least number of parameters except for the simple backbones, and the complexity of our model is comparable to the others in terms of FLOPs. In other words, the improvement of our method does come from network module design, rather than the increased model complexity.

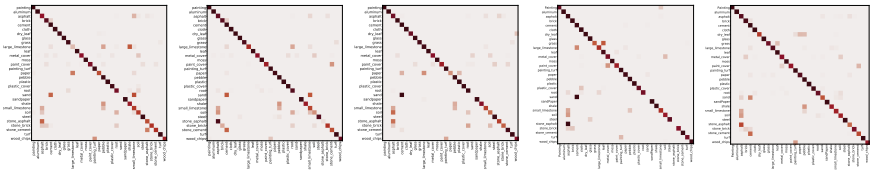
Training and running time comparison The training time and inference time of different methods are also compared in the Table 2, which are all evaluated on the same platform with a TITAN RTX GPU. The per-epoch training time and per-image testing time are reported. Both the training and test of DTPNet are faster than CLASSNet, but slower than other methods. One reason is that DTPNet requires cross-layer feature grouping and local 3D patch extraction. Such operations are currently not optimized for acceleration by Pytorch. How to accelerate them with some approximate operations with Pytorch-supported acceleration will be one of our future works.

4.3. Behavior Analysis

Ablation study We construct two baseline models for analyzing the effectiveness of the proposed DTP module as follows. *(a)* Model named 'w/o DTP' denotes a baseline model built by removing the DTP module in DTPNet. *(b)* Model named 'w/o GAP' denotes another baseline model from DTPNet by removing the GAP branch. *(c)* Model named 'Backbone^{+}' denotes the ResNet18 model but with more channels in each layer to maintain a similar number of parameters with DTPNet, *i.e.*, similar with 'w/o DTP' but more parameters. For a fair comparison, we train these models using the same schemes with the ResNet18 backbone used. The ablation results are summarized in Table 3. It can be seen that removing the DTP module brings a significant performance drop on both datasets, suggesting that DTP brings extra discriminability and robustness. In addition, the model without GAP performs slightly better than the model without DTP, indicating that our proposed DTP encoding works for texture recognition and even provides more benefits than only using the GAP branch. Both the results of model 'w/o DTP' and 'w/o GAP' are worse than the proposed DTPNet, which indicates that the DTP encoding and GAP features are crucial and complementary for texture recognition. Note that the proposed DTP encodes the fine-scale details

Table 3: Ablation study. ResNet18 backbone is used. The best results are marked in **bold**.

Dataset	w/o DTP	w/o GAP	Backbone$^+$	DTPNet
MINC	78.0 ± 0.6	79.2 ± 0.5	77.9 ± 0.5	**80.7** ± 0.4
GTOS-mobile	82.2 ± 1.3	82.5 ± 0.4	81.9 ± 1.5	**87.0** ± 1.2



(a) ResNet18 (b) DEPNet (c) CLASSNet (d) FENet (e) DTPNet

Figure 6: Confusion matrices of several models on GTOS-Mobile.

and the patterns in evolution, while the GAP of top-layer features captures coarse-scale semantics, which is a probable reason for the boosting with the combined feature. Finally, compared with 'Backbone$^+$', DTPNet shows significant improvements on both datasets, which indicates that the improvement of DTPNet does not come from the enlarged model size, but from its specific module design.

Confusion analysis For a deeper inspection of the behavior of DTPNet on each category, we compute the confusion matrices on the GTOS-Mobile dataset using ResNet18 backbone and compare it with ResNet backbone and other competitive methods, *i.e.* DEPNet [2], CLASSNet [17] and FENet [15]. The results are shown in Fig. 6 where the confusion matrices produced by DTPNet are more diagonally concentrated than other methods. It can also be seen that the proposed DTPNet can well distinguish the confusing pairs of FENet, *e.g.* Sand versus Cement and Stone Asphalt versus Asphalt. Such results further verify the stronger robustness and greater discriminative capacity provided by the proposed DTPNet.

Influences of local sampling strategies The proposed DTP module uses 3D ball sampling for local encoding. To further investigate the influences of local sampling strategies, we evaluate three DTPNet variants with different sampling patterns: 1) Cylinder, which uses uniform sampling circle size across layers. 2) Shrinking cone, which uses larger sampling circles in the preceding layers. 3) Dilating cone, which uses smaller sampling circles in the preceding layers. Only in-plane rotation is applied for code shifting in these variants. The models are all evaluated on the GTOS-mobile dataset using the ResNet18 backbone. The results are shown in Table 4. It can be seen that the other three sampling strategies result in around 1% drop in accuracy, which experimentally validates the superiority of the 3D ball scheme we used. Note that the proposed scheme enables circular shift among both the spatial domain and the scale pyramid, and retrieving the minimum value, which mimics the scheme for scale invariance in SIFT and SURF, *i.e.* finding maximum values in the spatial pyramid. This likely explains its superior performance over other sampling schemes.

Influences of local encoding schemes To investigate the influences of local en-

Table 4: Performance comparison between different sampling strategies on GTOS-M dataset.

Sampling	Cylinder	Shrinking Cone	Dilating Cone	3D ball (Ours)
DTPNet	85.9 ± 0.2	85.5 ± 0.4	85.8 ± 0.7	87.0 ± 1.2

Table 5: Performance comparison of different local encoding schemes. The best results are marked in **bold**. The ResNet18 backbone is used.

Encoding Scheme		DTD	KTH	MINC	GTOS-M
3D Statics	Std	71.5 ± 0.6	86.2 ± 2.2	80.7 ± 0.4	83.3 ± 0.3
	Mean	71.4 ± 0.8	86.5 ± 1.8	80.6 ± 0.6	84.0 ± 0.2
	Skewness	71.5 ± 0.6	86.2 ± 2.4	80.6 ± 0.4	84.9 ± 0.3
	Slope	71.5 ± 0.5	86.2 ± 2.1	80.4 ± 0.7	82.3 ± 0.3
	DBC	71.5 ± 0.4	85.4 ± 1.1	80.5 ± 0.6	85.3 ± 1.3
Handcrafted	SIFT	71.4 ± 0.5	86.3 ± 1.8	80.5 ± 0.5	85.9 ± 0.4
	SRI-LBP	71.5 ± 0.6	86.5 ± 1.6	80.6 ± 0.4	85.7 ± 0.1
Neural LBP	LBC	70.9 ± 0.3	84.5 ± 1.0	80.4 ± 0.5	83.0 ± 0.4
	LBoM	71.2 ± 0.4	85.5 ± 1.3	80.5 ± 0.6	84.7 ± 0.3
Ours	DTP	$\mathbf{71.8} \pm 0.7$	$\mathbf{86.7} \pm 1.3$	$\mathbf{80.7} \pm 0.4$	$\mathbf{87.0} \pm 1.2$

coding schemes for the cross-layer feature tensor, we select 5 commonly-used local statistics, in the place of DTP encoding. The results are shown in Table 5. Note that the differential box-counting (DBC), which is introduced in [17] for encoding the self-similarity statistics in the cross-layers of a CNN, is also evaluated and compared. As can be seen, all these cross-layer statistics benefit the classification performance over the backbone ResNet18, indicating that the implicit discriminative information in the cross-layer evolution does exist and benefit texture recognition. Along the presented statistics, the DTP brings the largest improvement in most cases, proving its effective capacity of dynamic evolution encoding for texture patterns.

We selected two handcrafted descriptors for further analysis, including the well-known SIFT descriptor [5] and an LBP variant called SRILBP [43], both of which are rotation and scale invariant. We integrate these descriptors into a ResNet18 backbone in a similar way as our DTPNet, forming two models denoted as SIFT-Net and SRILBP-Net, respectively. Their results are shown in Table 5. We can see that our DTPNet outperforms both models by over 1%, highlighting the advantages of the DTP module.

Furthermore, we also compared the proposed local encoding scheme with other existing LBP-inspired neural modules, namely LBC [32] and LBoM [33]. Baseline models are constructed by replacing the local encoding scheme with multiple LBC or LBoM layers. We trained the networks with the training parameters consistent with DTPNet. The results are shown in Table 5, where our DTPNet outperforms both baselines noticeably, demonstrating the effectiveness of our proposed scheme.

Performance on different backbones Using diferent backbone models may influence the performance of the DTPNet. Table 6 lists the results on DTD, KTH and FMD of DTPNet using ResNet18, ResNet50, ResNet101, and ResNet152 as the backbone. It can be seen that the deepest backbone ResNet152 gains the best accuracy, possibly due to that it contains more cross-layer dynamics for the DTP encoding.

Table 6: Performance comparison on different backbones.

Backbone	DTD	KTH	FMD
ResNet-18	71.8 ± 0.7	86.7 ± 1.3	85.7 ± 0.9
ResNet-50	73.5 ± 0.4	88.5 ± 1.6	87.8 ± 1.3
ResNet-101	74.0 ± 0.8	88.5 ± 2.0	87.9 ± 1.4
ResNet-152	74.5 ± 0.9	88.7 ± 1.6	88.5 ± 1.2

Table 7: Robustness test under four types of corruption. The ResNet18 backbone is used. Best results are in **bold**.

Corruption	Gaussian Noise (SNR)				Salt-pepper Noise (ρ)			
SNR / ρ	15	25	35	45	1%	2%	6%	10%
ResNet	58.06	77.94	80.20	80.53	64.80	58.47	41.49	28.60
DeepTEN	37.83	74.07	80.45	80.81	60.72	54.55	23.85	18.50
DEPNet	46.26	80.12	82.51	82.46	64.87	54.98	30.00	21.20
CLASSNet	62.88	81.87	85.39	85.62	71.00	65.89	47.10	31.64
FENet	64.36	83.88	85.69	85.87	69.68	62.56	46.88	38.33
DTPNet	**64.39**	**84.65**	**87.12**	**87.62**	**72.09**	**67.13**	**47.87**	**39.35**
Corruption	Poisson Noise (λ)				Gaussian Blur (σ)			
λ / σ	1	2	3	4	1	2	3	6
ResNet	74.12	67.52	63.12	59.58	79.99	63.02	54.53	29.69
DeepTEN	68.81	63.72	61.41	59.38	75.93	63.35	53.73	31.55
DEPNet	75.32	65.86	60.90	53.43	79.43	65.63	48.80	27.93
CLASSNet	79.51	74.53	70.28	66.85	80.79	69.78	54.68	28.16
FENet	79.72	74.98	69.54	65.66	81.14	70.05	58.57	29.56
DTPNet	**80.98**	**75.06**	**70.44**	**67.24**	**84.39**	**74.30**	**58.75**	**32.36**

Robustness analysis on different corruptions To further investigate the robustness of DTPNet, we conduct additional robustness tests against four types of corruption including 1) Gaussian noises with SNRs; 2) Poisson noises with different strengths λ; 3) Salt-pepper noises with different ratios ρ; and 4) Gaussian blurs using Gaussian kernels with different standard deviations σ. The results on the GTOS-Mobile dataset are shown in Table 7. All the methods experience declines in performance under the corruptions, particularly when these corruptions are severe. In contrast, the proposed method maintains relatively higher performance under all the corruptions, indicating robustness against diverse distortions.

Visualizing DTP encoding To verify the effectiveness of DTP encoding, we illustrate the DTP codes on four texture images within two classes, see Fig. 7. As can be seen in the figure, two texture patterns from the same class but in different rotated angles contribute to a similar representation, *e.g.* (a) v.s. (b), indicating the rotation invariance brought by the robust DTP descriptor on the features. Besides, two texture patterns from the same class but with different colors devote to a similar representation, *e.g.* (c) v.s. (d), demonstrating the illumination invariance brought by the proposed method. The DTP codes on different classes of texture images significantly differ, *e.g.* (a/b) v.s. (c/d), which demonstrates the discriminative capacity of DTP module.

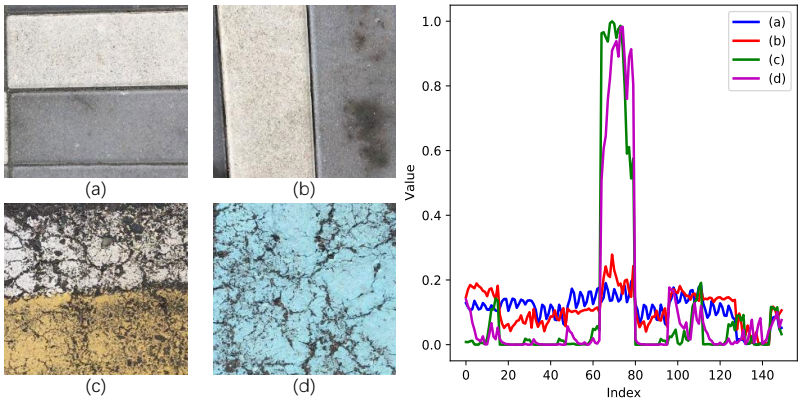


Figure 7: Illustration of the flattened feature tensors produced by the DTP module. Left: the input images. Image (a) and (b) belong to class "Brick", while image (c) and (d) belong to another class "Painting". Right: the plot of the global feature vectors.

Figure 8: Confusing cases of DTPNet on GTOS-Mobile. Top: failure samples and their true labels. Bottom: The incorrectly-predicted labels and samples from the predicted classes.

Limitation and discussion To investigate the limitations of the proposed DTPNet, we further show some confusing cases in Fig. 8. It can be seen that each confusing pair has quite a similar appearance with only slight differences in the fine-grained patterns, leading to great challenges for discrimination. Note that such confusing cases are also challenging for other SOTA methods, *e.g.* CLASSNet. To address these confusing cases, it might be necessary to develop a module that captures the subtle discrepancies, which will be considered in our future work.

5. Conclusion

The paper makes an effort on improving local encoding for texture recognition in the CNN period. The proposed DTP encoding mechanism can well encode the fine-scale details and CNN-evolutional patterns in local codes, with rotation invariance originating from the LBP-based coding. Extensive experiments demonstrated

the effectiveness of the proposed DTP mechanism. LBP and its variants are deeply researched in the past. Besides the rotation-invariant design, other well-established designs, *e.g.* equivalence-class, and ternary codes, also have the potential in enhancing local features in CNNs, which will be our future work. In addition, texture representation is a fundamental task in computer vision and thus can undoubtedly benefit other vision applications, *e.g.*, object recognition, scene recognition, fine-grained recognition, semantic segmentation and texture synthesis. Therefore, we would like to extend our idea of encoding cross-layer feature tensor patches via invariant local pattern encoding to other image classification tasks.

Acknowledgments

Yuhui Quan would like to acknowledge the partial support from Fundamental Research Funds for the Central Universities of China under Grant x2jsD2230220, the partial support from National Natural Science Foundation of China under Grant 62372186, and the partial support from Natural Science Foundation of Guangdong Province, China under Grants 2022A1515011755 and 2023A1515012841. Ruotao Xu would like to acknowledge the partial support from National Natural Science Foundation of China under Grant 62106077 and the partial support Natural Science Foundation of Guangdong Province, China under Grant 2022A1515011087. Lianwen Jin would like to thank the partial support from Zhuhai Industry Core and Key Technology Research Project, China under Grant 2220004002350. Yong Xu would like to thank the partial support from National Natural Science Foundation of China under Grant 62072188 and the partial support from State Administration of Foreign Experts Affairs, China under Grant G2023163015L.

References

[1] S. Bell, P. Upchurch, N. Snavely, K. Bala, Material recognition in the wild with the materials in context database, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015.

[2] J. Xue, H. Zhang, K. Dana, Deep texture manifold for ground terrain recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 558–567.

[3] C. Ding, J. Choi, D. Tao, L. S. Davis, Multi-directional multi-level dual-cross patterns for robust face recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 38 (3) (2015) 518–531.

[4] L. Liu, J. Chen, P. Fieguth, G. Zhao, R. Chellappa, M. Pietikäinen, From BoW to CNN: Two Decades of Texture Representation for Texture Classification, International Journal of Computer Vision 127 (1) (2019) 74–109.

[5] D. G. Lowe, Distinctive image features from scale-invariant keypoints, International Journal of Computer Vision 60 (2) (2004) 91–110.

[6] T. Ojala, M. Pietikainen, T. Maenpaa, Multiresolution gray-scale and rotation invariant texture classification with local binary patterns, IEEE Transactions on pattern analysis and machine intelligence 24 (7) (2002) 971–987.

[7] S. Lazebnik, C. Schmid, J. Ponce, A sparse texture representation using local affine regions, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (8) (2005) 1265–1278.

[8] Y. Xu, H. Ji, C. Fermüller, Viewpoint invariant texture description using fractal analysis, International Journal of Computer Vision 83 (1) (2009) 85–100.

[9] V. Andrearczyk, P. F. Whelan, Using filter banks in convolutional neural networks for texture classification, Pattern Recognition LETT 84 (2016) 63–69.

[10] S. Fujieda, K. Takayama, T. Hachisuka, Wavelet convolutional neural networks for texture classification, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (Jul 2017).

[11] F. Perronnin, C. Dance, Fisher kernels on visual vocabularies for image categorization, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2007, pp. 1–8.

[12] H. Zhang, J. Xue, K. Dana, Deep ten: Texture encoding network, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 2896–2905.

[13] T. Mäenpää, M. Pietikäinen, Multi-scale binary patterns for texture analysis, in: J. Bigun, T. Gustavsson (Eds.), Image Analysis, Springer Berlin Heidelberg, Berlin, Heidelberg, 2003, pp. 885–892.

[14] L. Van der Maaten, G. Hinton, Visualizing data using t-sne, Journal of machine learning research 9 (11) (2008).

[15] Y. Xu, F. Li, Z. Chen, J. Liang, Y. Quan, Encoding spatial distribution of convolutional features for texture representation, Advances in Neural Information Processing Systems 34 (2021) 22732–22744.

[16] S. Mao, D. Rajan, L. T. Chia, Deep residual pooling network for texture recognition, Pattern Recognition 112 (2021) 107817.

[17] Z. Chen, F. Li, Y. Quan, Y. Xu, H. Ji, Deep texture recognition via exploiting cross-layer statistical self-similarity, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 5231–5240.

[18] L. Liu, J. Chen, P. Fieguth, G. Zhao, R. Chellappa, M. Pietikäinen, From bow to cnn: Two decades of texture representation for texture classification, International Journal of Computer Vision (Nov) (2018) 1–36.

[19] B. Caputo, E. Hayman, P. Mallikarjuna, Class-specific material categorisation, in: Proceedings of the IEEE International Conference on Computer Vision, Vol. 2, 2005, pp. 1597 – 1604 Vol. 2.

[20] Z. Guo, L. Zhang, D. Zhang, A completed modeling of local binary pattern operator for texture classification, IEEE Transactions on Image Processing 19 (6) (2010) 1657–1663.

[21] H. Jegou, M. Douze, C. Schmid, P. Perez, Aggregating local descriptors into a compact image representation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2010.

[22] J. Bruna, S. Mallat, Invariant scattering convolution networks, IEEE Transactions on Pattern Analysis and Machine Intelligence 35 (2013) 1872–1886.

[23] M. Cimpoi, S. Maji, A. Vedaldi, Deep filter banks for texture recognition and segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3828–3836.

[24] T.-Y. Lin, S. Maji, Visualizing and understanding deep texture representations, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2791–2799.

[25] X. Bu, Y. Wu, Z. Gao, Y. Jia, Deep convolutional network with locality and sparsity constraints for texture classification, Pattern Recognition 91 (2019) 34–46.

[26] W. Zhai, Y. Cao, Z. Zha, H. Xie, F. Wu, Deep structure-revealed network for texture recognition, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, IEEE, 2020, pp. 11007–11016.

[27] Y. Hu, Z. Long, G. AlRegib, Multi-level texture encoding and representation (multer) based on deep neural networks, in: IEEE International Conference on Image Processing, 2019.

[28] Q. Luo, J. Su, C. Yang, O. Silven, L. Liu, Scale-selective and noise-robust extended local binary pattern for texture classification, Pattern Recognition (2022) 108901.

[29] J. Tang, Q. Su, B. Su, S. Fong, W. Cao, X. Gong, Parallel ensemble learning of convolutional neural networks and local binary patterns for face recognition, Computer Methods and Programs in Biomedicine 197 (2020) 105622.

[30] L. Liu, S. Lao, P. W. Fieguth, Y. Guo, X. Wang, M. Pietikäinen, Median Robust Extended Local Binary Pattern for Texture Classification, IEEE Transactions on Image Processing 25 (3) (2016) 1368–1381.

[31] X. Shu, H. Pan, J. Shi, X. Song, X.-J. Wu, Using global information to refine local patterns for texture representation and classification, Pattern Recognition 131 (2022) 108843.

[32] F. Juefei-Xu, V. Naresh Boddeti, M. Savvides, Local binary convolutional neural networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 19–28.

[33] X. Zhang, L. Liu, Y. Xie, J. Chen, L. Wu, M. Pietikainen, Rotation invariant local binary convolution neural networks, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 1210–1219.

[34] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.

[35] K. He, X. Zhang, S. Ren, J. Sun, Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1026–1034.

[36] J. Xue, H. Zhang, K. Dana, K. Nishino, Differential angular imaging for material recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 764–773.

[37] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, A. Vedaldi, Describing textures in the wild, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 3606–3613.

[38] L. Sharan, R. Rosenholtz, E. Adelson, Material perception: What can you see in a brief glance?, Journal of Vision 9 (2010) 784–784.

[39] W. Zhai, Y. Cao, J. Zhang, Z. Zha, Deep multiple-attribute-perceived network for real-world texture recognition, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, IEEE, 2019, pp. 3612–3621.

[40] J. Peeples, W. Xu, A. Zare, Histogram layers for texture analysis (2020). arXiv:2001.00215.

[41] V. Pandey, M. Gubba, M. Faisal, T. Kalra, Ensembling framework for texture extraction techniques for classification, arXiv preprint arXiv:2206.04158 (2022).

[42] K. Song, H. Yang, Z. Yin, Multi-scale boosting feature encoding network for texture recognition, IEEE Transactions on Circuits and Systems for Video Technology 31 (11) (2021) 4269–4282.

[43] R. Davarzani, S. Mozaffari, K. Yaghmaie, Scale-and rotation-invariant texture description with improved local binary pattern features, Signal Processing 111 (2015) 274–293.