International Journal of Computer Vision
https://doi.org/10.1007/s11263-018-1125-z

From BoW to CNN: Two Decades of Texture Representation for Texture Classification

Li Liu[1,2] · Jie Chen[2] · Paul Fieguth[3] · Guoying Zhao[2] · Rama Chellappa[4] · Matti Pietikäinen[2]

Received: 6 January 2018 / Accepted: 6 October 2018

Abstract
Texture is a fundamental characteristic of many types of images, and texture representation is one of the essential and challenging problems in computer vision and pattern recognition which has attracted extensive research attention over several decades. Since 2000, texture representations based on Bag of Words and on Convolutional Neural Networks have been extensively studied with impressive performance. Given this period of remarkable evolution, this paper aims to present a comprehensive survey of advances in texture representation over the last two decades. More than 250 major publications are cited in this survey covering different aspects of the research, including benchmark datasets and state of the art results. In retrospect of what has been achieved so far, the survey discusses open challenges and directions for future research.

Keywords Texture classification · Feature extraction · Deep learning · Local descriptors · Bag of Words · Computer vision · Visual attributes · Convolutional Neural Network

1 Introduction

Our visual world is richly filled with a great variety of textures, present in images ranging from multispectral satellite data to microscopic images of tissue samples (see Fig. 1). As a powerful visual cue, like color, texture provides useful information in identifying objects or regions of interest in images. Texture is different from color in that it refers to the spatial organization of a set of basic elements or primitives (i.e., textons), the fundamental microstructures in natural images and the atoms of preattentive human visual perception (Julesz 1981). A textured region will obey some statistical properties, exhibiting periodically repeated textons with some degree of variability in their appearance and relative position (Forsyth and Ponce 2012). Textures may range from purely stochastic to perfectly regular and everything in between (see Fig. 1).

As a longstanding, fundamental and challenging problem in the fields of computer vision and pattern recognition, texture analysis has been a topic of intensive research since the 1960s (Julesz 1962) due to its significance both in understanding how the texture perception process works in human vision as well as in the important role it plays in a wide variety of applications. The analysis of texture traditionally embraces several problems including classification, segmentation, synthesis and shape from texture (Tuceryan and Jain 1993). Significant progress has been made since the 1990s in the first three areas, with shape from texture receiving comparatively less attention. Typical applications of texture analysis include medical image analysis (Depeursinge et al. 2017; Nanni et al. 2010; Peikari et al. 2016), quality inspection (Xie and Mirmehdi 2007), content based image retrieval (Manjunath and Ma 1996; Sivic and Zisserman 2003; Zheng et al. 2018), analysis of satellite or aerial imagery (Kan-

Communicated by Xiaoou Tang.

✉ Li Liu
li.liu@oulu.fi

Jie Chen
jie.chen@oulu.fi

Paul Fieguth
pfieguth@uwaterloo.ca

Guoying Zhao
guoying.zhao@oulu.fi

Rama Chellappa
rama@umiacs.umd.edu

Matti Pietikäinen
matti.pietikainen@oulu.fi

1 National University of Defense Technology, Changsha, China

2 University of Oulu, Oulu, Finland

3 University of Waterloo, Waterloo, Canada

4 University of Maryland, College Park, USA

Published online: 08 November 2018

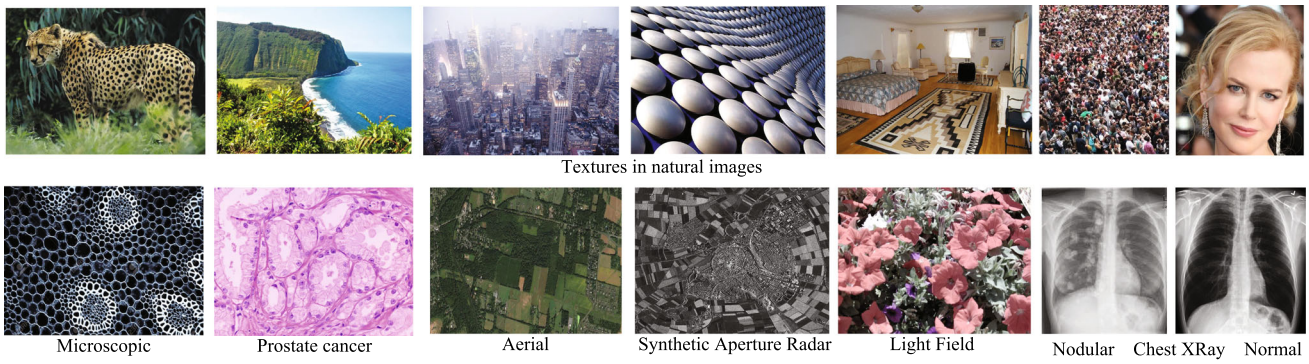


Fig. 1 Texture is an important characteristic of many types of images

daswamy et al. 2005; He et al. 2013), face analysis (Ahonen et al. 2006b; Ding et al. 2016; Simonyan et al. 2013; Zhao and Pietikäinen 2007), biometrics (Ma et al. 2003; Pietikäinen et al. 2011), object recognition (Shotton et al. 2009; Oyallon and Mallat 2015; Zhang et al. 2007), texture synthesis for computer graphics and image compression (Gatys et al. 2015, 2016), and robot vision and autonomous navigation for unmanned aerial vehicles. The ever-increasing amount of image and video data due to surveillance, handheld devices, medical imaging, robotics etc. offers an endless potential for further applications of texture analysis.

Texture representation, i.e., the extraction of features that describe texture information, is at the core of texture analysis. After over five decades of continuous research, many kinds of theories and algorithms have emerged, with major surveys and some representative work as follows. The majority of texture features before 1990 can be found in surveys and comparative studies (Conners and Harlow 1980; Haralick 1979; Ohanian and Dubes 1992; Reed and Dubuf 1993; Tuceryan and Jain 1993; Van Gool et al. 1985; Weszka et al. 1976). Tuceryan and Jain (1993) identified five major categories of features for texture discrimination: statistical, geometrical, structural, model based, and filtering based features. Ojala et al. (1996) carried out a comparative study to evaluate the classification performance of several texture features. Randen and Husoy (1999) reviewed most major filtering based texture features and performed a comparative performance evaluation for texture segmentation. Zhang and Tan (2002) reviewed invariant texture feature extraction methods. Zhang et al. (2007) evaluated the performance of several major invariant local texture descriptors. The 2008 book "Handbook of Texture Analysis" edited by Mirmehdi et al. (2008) contains representative work on texture analysis—from 2D to 3D, from feature extraction to synthesis, and from texture image acquisition to classification. The book "Computer Vision Using Local Binary Patterns" by Pietikäinen et al. (2011) provides an excellent overview of the theory of Local Binary Patterns (LBP) and the use in solving various kinds of problems in computer vision, especially in biomedical applications and biometric recognition systems. Huang et al. (2011) presented a review of the LBP variants in the application area of facial image analysis. The book "Local Binary Patterns: New Variants and Applications" by Brahnam et al. (2014) is a collection of several new LBP variants and their applications to face recognition. More recently, Liu et al. (2017) conducted a taxonomy of recent LBP variants and performed a large scale performance evaluation of forty texture features. Researchers (Raad et al. 2017; Akl et al. 2018) presented a review of exemplar based texture synthesis approaches.

The published surveys (Conners and Harlow 1980; Haralick 1979; Ohanian and Dubes 1992; Reed and Wechsler 1990; Reed and Dubuf 1993; Ojala et al. 1996; Pichler et al. 1996; Tuceryan and Jain 1993; Van Gool et al. 1985) mainly reviewed or compared methods prior to 1995. Similarly, the articles (Randen and Husoy 1999; Zhang and Tan 2002) only covered approaches before 2000. There are more recent surveys (Brahnam et al. 2014; Huang et al. 2011; Liu et al. 2017; Pietikäinen et al. 2011), however they focused exclusively on texture features based on LBP. The emergence of many powerful texture analysis techniques has given rise to a further increase in research activity in texture research since 2000, however none of these published surveys provides an extensive survey over that time. Given recent developments, we believe that there is a need for an updated survey, motivating this present work. A thorough review and survey of existing work, the focus of this paper, will contribute to more progress in texture analysis. Our goal is to overview the core tasks and key challenges in texture representation approaches, to define taxonomies of representative approaches, to provide a review of texture datasets, and to summarize the performance of the state of the art on publicly available datasets. According to the different visual representations, this survey categorizes the texture representation literature into three broad types: Bag of Words (BoW)-based, Convolutional Neural Network (CNN)-based, and attribute-based. The BoW-based methods are organized according to their key components. The CNN-based methods are categorized into one of pretrained CNN

models, finetuned CNN models, or handcrafted deep convolutional networks.

The remainder of this paper is organized as follows. Related background, including the problem and its applications, the progress made during the past decades, and the challenges of the problem, are summarized in Sect. 2. From Sects. 3 to 5 we give a detailed review of texture representation techniques for texture classification by providing a taxonomy to more clearly group the prominent alternatives. A summarization of benchmark texture databases and state of the art performance is given in Sect. 6. Section 7 concludes the paper with a discussion of promising directions for texture representation.

2 Background

2.1 The Problem

Texture analysis can be divided into four areas: classification, segmentation, synthesis, and shape from texture (Tuceryan and Jain 1993). Texture classification (Lazebnik et al. 2005; Liu and Fieguth 2012; Tuceryan and Jain 1993; Varma and Zisserman 2005, 2009) deals with designing algorithms for declaring a given texture region or image as belonging to one of a set of known texture categories of which training samples have been provided. Texture classification may also be a binary hypothesis testing problem, such as differentiating a texture as being within or outside of a given class, such as distinguishing between healthy and pathological tissues in medial image analysis. The goal of texture segmentation is to partition a given image into disjoint regions of homogeneous texture (Jain and Farrokhnia 1991; Manjunath and Chellappa 1991; Reed and Wechsler 1990; Shotton et al. 2009). Texture synthesis is the process of generating new texture images which are perceptually equivalent to a given texture sample (Efros and Leung 1999; Gatys et al. 2015; Portilla and Simoncelli 2000; Raad et al. 2017; Wei and Levoy 2000; Zhu et al. 1998). As textures provide powerful shape cues, approaches for shape from texture attempt to recover the three dimensional shape of a textured object from its image. It should be noted that the concept of "texture" may have different connotations or definitions depending on the given objective. Classification, segmentation, and synthesis are closely related and widely studied, with shape from texture receiving comparatively less attention. Nevertheless, texture representation is at the core of these four problems. Texture representation, together with texture classification, will form the primary focus of this survey.

As a classical pattern recognition problem, texture classification primarily consists of two critical subproblems: texture representation and classification (Jain et al. 2000). It is generally agreed that the extraction of powerful texture features plays a relatively more important role, since if poor features are used even the best classifier will fail to achieve good results. While this survey is not explicitly concerned with texture synthesis, studying synthesis can be instructive, for example, classification of textures via *analysis by synthesis* (Gatys et al. 2015) in which a model is first constructed for synthesizing textures and then inverted for the purposes of classification. As a result, we will include representative texture modeling methods in our discussion.

2.2 Summary of Progress in the Past Decades

Milestones in texture representation over the past decades are listed in Fig. 2. The study of texture analysis can be traced back to the earliest work of Julesz (1962), who studied the theory of human visual perception of texture and suggested that texture might be modelled using kth order statistics—the cooccurrence statistics for intensities at k-tuples of pixels. Indeed, early work on texture features in the 1970s, such as the well known Gray Level Cooccurrence Matrix (GLCM) method (Haralick et al. 1973; Haralick 1979), were mainly driven by this perspective. Aiming at seeking essential ingredients in terms of features and statistics in human texture perception, in the early 1980s Julesz (1981), Julesz and Bergen (1983) proposed the texton theory to explain texture preattentive discrimination, which states that textons (composed of local conspicuous features such as corners, blobs, terminators and crossings) are the elementary units of preattentive human texture perception and only the *first* order statistics of textons have perceptual significance: textures having the same texton densities could not be discriminated. Julesz's texton theory has been widely studied and has largely influenced the development of texture analysis methods.

Research on texture features in the late 1980s and the early 1990s mainly focused on two well-established areas:

1. *Filtering* approaches, which convolve an image with a bank of filters followed by some nonlinearity. One pioneering approach was that of Laws (1980), where a bank of separable filters was applied, with subsequent filtering methods including Gabor filters (Bovik et al. 1990; Jain and Farrokhnia 1991; Turner 1986), Gabor wavelets (Manjunath and Ma 1996), wavelet pyramids (Freeman and Adelson 1991; Mallat 1989), and simple linear filters like Differences of Gaussians (Malik and Perona 1990).
2. *Statistical modelling*, which characterizes texture images as arising from probability distributions on random fields, such as a Markov Random Field (MRF) (Cross and Jain 1983; Mao and Jain 1992; Chellappa and Chatterjee 1985; Li 2009) or fractal models (Keller et al. 1989; Mandelbrot and Pignoni 1983).

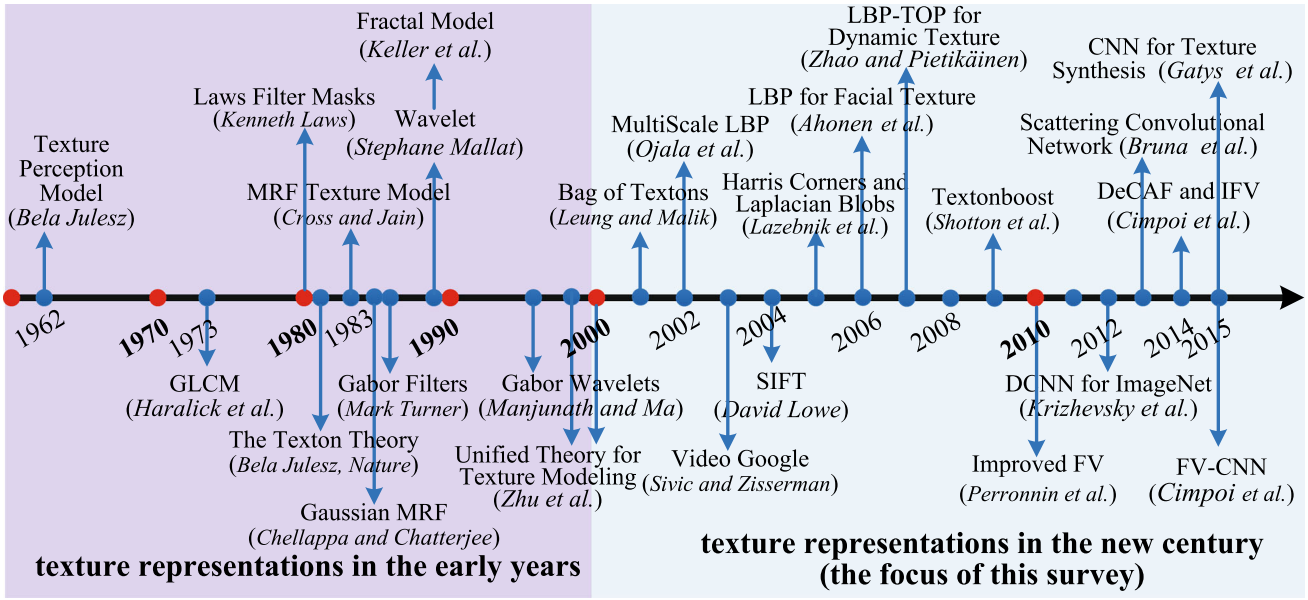


Fig. 2 The evolution of texture representation over the past decades (see discussion in Sect. 2.2)

At the end of the last century there was a renaissance of texton-based approaches, including Wu et al. (2000); Xie et al. (2015); Zhu et al. (1998, 2000, 2005); Zhu (2003) on the mathematical modelling of textures and textons. A notable stride was the Bag of Textons (BoT) (Leung and Malik 2001) and later Bag of Words (BoW) (Csurka et al. 2004; Sivic and Zisserman 2003; Vasconcelos and Lippman 2000) approaches, where a dictionary of textons is generated, and images are represented statistically as orderless histograms over the texton dictionary.

In the 1990s, the need for invariant feature representations was recognized, to reduce or eliminate sensitivity to variations such as illumination, scale, rotation, view point etc. This gave rise to the development of local invariant descriptors, particularly milestone texture features such as Scale Invariant Feature Transform (SIFT) (Lowe 2004), Speeded Up Robust Features (SURF) (Bay et al. 2006) and LBP (Ojala et al. 2002b). Such local handcrafted texture descriptors dominated many domains of computer vision until the turning point in 2012 when deep Convolutional Neural Networks (CNN) (Krizhevsky et al. 2012) achieved record-breaking image classification accuracy. Since that time the research focus has been on deep learning methods for many problems in computer vision, including texture analysis (Cimpoi et al. 2014, 2015, 2016).

The importance of texture representations [such as Gabor filters (Manjunath and Ma 1996), LBP (Ojala et al. 2002b), BoT (Leung and Malik 2001), Fisher Vector (FV) (Sanchez et al. 2013), and wavelet Scattering Convolution Networks (ScatNet) (Bruna and Mallat 2013)] is that they were found to be well applicable to other problems of image understanding and computer vision, such as object recognition (Everingham et al. 2015; Russakovsky et al. 2015), scene classification (Bosch et al. 2008; Cimpoi et al. 2016; Kwitt et al. 2012; Renninger and Malik 2004) and facial image analysis (Ahonen et al. 2006a; Simonyan et al. 2013; Zhao and Pietikäinen 2007). For instance, recently many of the best object recognition approaches in challenges such as PASCAL VOC (Everingham et al. 2015) and ImageNet ILSVRC (Russakovsky et al. 2015) were based on variants of texture representations. Beyond BoT (Leung and Malik 2001) and FV (Sanchez et al. 2013), researchers developed Bag of Semantics (BoS) (Dixit et al. 2015; Dixit and Vasconcelos 2016; Kwitt et al. 2012; Li et al. 2014; Rasiwasia and Vasconcelos 2012) which requires classifying image patches using BoT or CNN and considers the class posterior probability vectors as locally extracted semantic descriptors. On the other hand, texture representations optimized for objects were also found to perform well for texture-specific problems (Cimpoi et al. 2014, 2015, 2016). As a result, the division between texture descriptors and more generic image or video descriptors has been narrowing. The study of texture representation continues to play an important role in computer vision and pattern recognition.

2.3 Key Challenges

In spite of several decades of development, most texture features have not been capable of performing at a level sufficient for real-world textures and are computationally too complex to meet the real-time requirements of many computer vision applications. The inherent difficulty in obtaining pow-

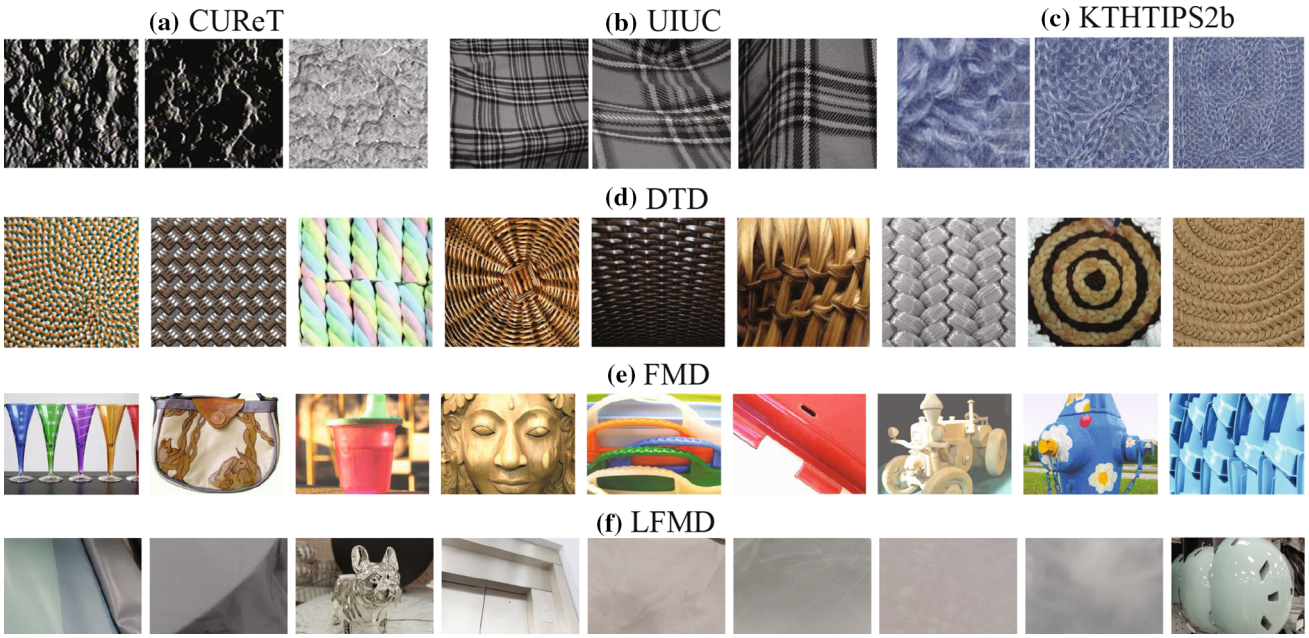


Fig. 3 Illustrations of challenges in texture recognition. Dramatic intraclass variations: **a** illumination variations, **b** view point and local nonrigid deformation, **c** scale variations, and **d** different instances from the same category. Small interclass variations make the problem harder still: **e** images from the FMD database, and **f** images from the LFMD database (photographed with a light-field camera). The reader is invited to identify the material category of the foreground surfaces in each image in (**e**, **f**). The correct answers are (from left to right): **e** glass, leather, plastic, wood, plastic, metal, wood, metal and plastic; **f** leather, fabric, metal, metal, paper, leather, water, sky and plastic. Sect. 6 gives details regarding texture databases

erful texture representations lies in balancing two competing goals: *high quality representation* and *high efficiency*.

High Quality related challenges mainly arise due to the large intraclass appearance variations caused by changes in illumination, rotation, scale, blur, noise, occlusion, etc. and potentially small interclass appearance differences, requiring texture representations to be of high robustness and distinctiveness. Illustrative examples are shown in Fig. 3. A further difficulty is in obtaining sufficient training data in the form of labeled examples, which are frequently available only in limited amounts due to collection time or cost.

High Efficiency related challenges include the potentially large number of different texture categories and their high dimensional representations. Here we have polar opposite motivations: that of big data, with associated grand challenges and the scalability/complexity of huge problems, and that of tiny devices, the growing need for deploying highly compact and efficient texture representations on resource-limited platforms such as embedded and handheld devices.

3 Bag of Words based Texture Representation

The goal of texture representation or texture feature extraction is to transform the input texture image into a feature vector that describes the properties of a texture, facilitating subsequent tasks such as texture classification, as illustrated in Fig. 4. Since texture is a spatial phenomenon, texture representation cannot be based on a single pixel, and generally requires the analysis of patterns over local pixel neighborhoods. Therefore, a texture image is first transformed to a pool of local features, which are then aggregated into a global representation for an entire image or region. Since the properties of texture are usually translationally invariant, most texture representations are based on an orderless aggregation of local texture features, such as a sum or max operation.

Early in 1981, Julesz (1981) introduced "textons", which refer to basic image features such as elongated blobs, bars, crosses, and terminators, as the elementary units of preattentive human texture perception. However Julesz's texton studies were limited by their exclusive focus on artificial texture patterns rather than natural textures. In addition, Julesz did not provide a rigorous definition for textons. Subsequently, texton theory fell into disfavor as a model of texture discrimination until the influential work by Leung and Malik (2001) who revisited textons and gave an operational definition of a texton as a cluster center in filter response space. This not only enabled textons to be generated automatically from an image, but also opened up the possibility of learning a universal texton dictionary for all images. Texture images can be statistically represented as histograms over a texton

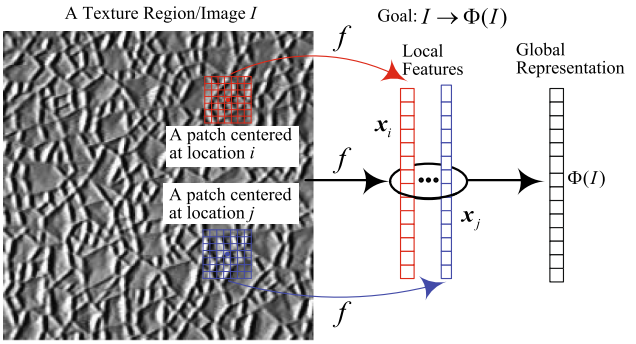


Fig. 4 The goal of texture representation is to transform the input texture image into a feature vector that describes the properties of the texture, facilitating subsequent tasks such as texture recognition. Usually a texture image is first transformed into a pool of local features, which are then aggregated into a global representation for an entire image or region

dictionary, referred to as the Bag of Textons (BoT) approach. Although BoT was initially developed in the context of texture recognition (Leung and Malik 2001; Malik et al. 1999), it was introduced/generalized to image retrieval (Sivic and Zisserman 2003) and classification (Csurka et al. 2004), where it was referred to as Bag of Features (BoF) or, more commonly, Bag of Words (BoW). The research community has since witnessed the prominence of the BoW model for over a decade during which many improvements were proposed.

3.1 The BoW Pipeline

The BoW pipeline is sketched in Fig. 5, consisting of the following basic steps:

1. Local Patch Extraction For a given image, a pool of N image patches is extracted over a sparse set of points of interest (Lazebnik et al. 2005; Zhang et al. 2007), over a fixed grid (Kong and Wang 2012; Marszałek et al. 2007; Sharan et al. 2013), or densely at each pixel position (Ojala et al. 2002b; Varma and Zisserman 2005, 2009).

2. Local Patch Representation Given the extracted N patches, local texture descriptors are applied to obtain a set or pool of texture features of D dimension. We denote the local features of N patches in an image as $\{\boldsymbol{x}_i\}_{i=1}^{N}, \boldsymbol{x}_i \in \mathbb{R}^D$. Ideally, local descriptors should be distinctive and at the same time robust to a variety of possible image transformations, such as scale, rotation, blur, illumination, and viewpoint changes. High quality local texture descriptors play a critical role in the BoW pipeline.

3. Codebook Generation The objective of this step is to generate a codebook (i.e., a texton dictionary) with K codewords $\{\boldsymbol{w}_i\}_{i=1}^{K}$, $\boldsymbol{w}_i \in \mathbb{R}^D$ based on training data. The codewords may be learned [e.g., by $k$means (Lazebnik et al. 2003; Varma and Zisserman 2005)] or in a predefined way [such as LBP (Ojala et al. 2002b)]. The size and nature of the codebook affects the representation followed and thus the discrimination power. The key here is how to generate a compact and discriminative codebook so as to enable accurate and efficient classification.

4. Feature Encoding Given the generated codebook and the extracted local texture features $\{\boldsymbol{x}_i\}$ from an image, feature encoding represents each local feature $\boldsymbol{x}_i$ with the codebook, usually by mapping each $\boldsymbol{x}_i$ to one or a number of codewords, resulting a feature coding vector $\boldsymbol{v}_i$ (e.g. $\boldsymbol{v}_i \in \mathbb{R}^K$). Of all the steps in the BoW pipeline, feature encoding is a core component which links local representation

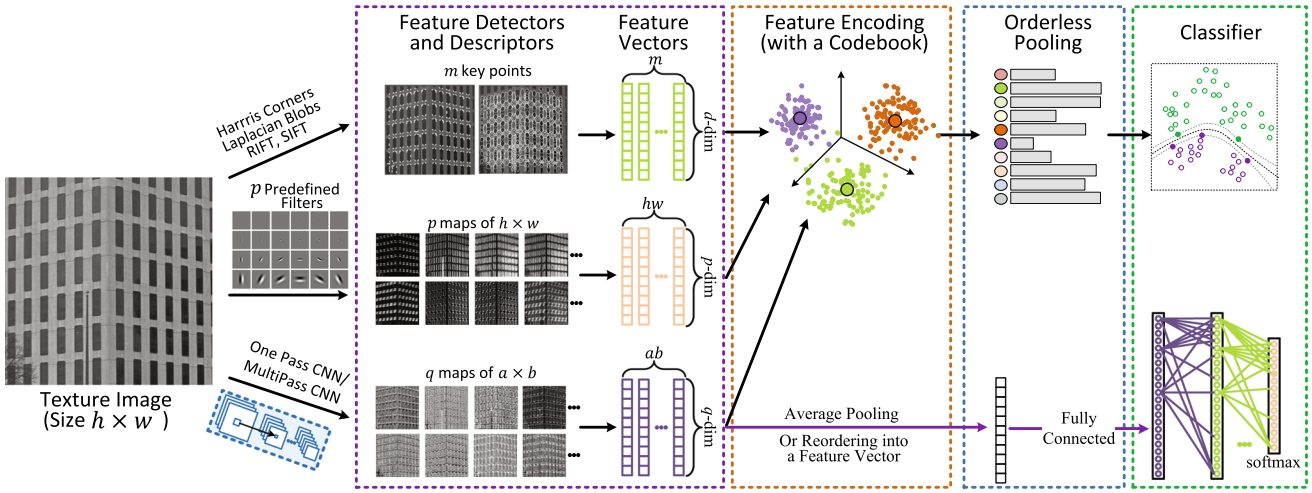


Fig. 5 General pipeline of the BoW model. See Table 1, and also refer to Sect. 3 for detail discussion. Features are computed from handcrafted detectors for descriptors like SIFT and RIFT, and densely applied local texture descriptors like handcrafted filters or CNNs. The CNN features can also be computed in an end-to-end manner using finetuned CNN models. These local features are quantized to visual words in a codebook

and feature pooling, greatly influencing texture classification in terms of both accuracy and speed. Thus, many studies have focused on developing powerful feature encoding, such as vector quantization/$k$means, sparse coding (Mairal et al. 2008, 2009; Peyré 2009), Locality constrained Linear Coding (LLC) (Wang et al. 2010), Vector of Locally Aggregated Descriptors (VLAD) (Jegou et al. 2012), and Fisher Vector (FV) (Cimpoi et al. 2016; Perronnin et al. 2010; Sanchez et al. 2013).

5. Feature Pooling A global feature representation $\boldsymbol{y}$ is produced by using a feature pooling strategy to aggregate the coded feature vectors $\{\boldsymbol{v}_i\}$. Classical pooling methods include average pooling, max pooling, and Spatial Pyramid Pooling (SPM) (Lazebnik et al. 2006; Timofte and Van Gool 2012).

6. Feature Classification The global feature is used as the basis for classification, for which many approaches are possible (Jain et al. 2000; Webb and Copsey 2011): Nearest Neighbor Classifier (NNC), Support Vector Machines (SVM), neural networks, and random forests. SVM is one of the most widely used classifiers for the BoW based representation.

The remainder of this section will introduce the methods in each component, as summarized in Table 1.

3.2 Local Texture Descriptors

All local texture descriptors aim to provide local representations invariant to contrast, rotation, scale, and possibly other criteria. The primary categorization is whether the descriptor is applied densely, at every pixel, as opposed to sparsely, only at certain locations of interest.

3.2.1 Sparse Texture Descriptors

To develop a sparse texture descriptor, a region of interest detector must be designed which is able to reliably detect a sparse set of regions, reliably and stably, under various imaging conditions. Typically, the detected regions undergo a geometric normalization, after which local descriptors are applied to encode the image content. A series of region detectors and local descriptors has been proposed, with excellent surveys (Mikolajczyk and Schmid 2005; Mikolajczyk et al. 2005; Tuytelaars et al. 2008). The sparse approach was introduced to texture recognition by Lazebnik et al. (2003), Lazebnik et al. (2005) and followed by Zhang et al. (2007).

In (Lazebnik et al. 2005) two types of complementary region detectors, the Harris affine detector of Mikolajczyk and Schmid (2002) and the Laplacian blob detector of Gårding and Lindeberg (1996), were used to detect affine covariant regions, meaning that the region content is affine invariant. Each detected region can be thought of as a texture element having a characteristic elliptic shape and a distinctive appearance pattern. In order to achieve affine invariance, each elliptical region was normalized and then two rotation invariant descriptors, the spin image (SPIN) and the Rotation Invariant Feature Transform (RIFT) descriptor, were applied. As a result, for each texture image four feature channels were extracted (two detectors $\times$ two descriptors), and for each feature channel $k$means clustering is performed to form its signature. The Earth Mover's Distance (EMD) (Rubner et al. 2000) was used for measuring the similarity between image signatures and NNC was used for classification. The Harris affine regions and Laplacian blobs in combination with SPIN and RIFT descriptors (i.e. the (H+L)(S+R) method) have demonstrated good performance (listed in Table 4) in classifying textures with significant affine variations, evidenced by the classification rate 96.0% on UIUC with a NNC classifier. Although this approach achieve affine invariance, they lack distinctiveness since some spatial information is lost due to their feature pooling schemes.

Following Lazebnik et al. (2005), Zhang et al. (2007) presented an evaluation of multiple region detector types, levels of geometric invariance, multiple local texture descriptors, and SVM classifier with kernels based on two effective measures for comparing distributions (signatures and EMD distance vs. standard BoW and the Chi Square distance) for texture and object recognition. Regarding local description, Zhang et al. (2007) also used the SIFT descriptor[1] in addition to SPIN and RIFT. With SVM classification, Zhang et al. (2007) showed significant performance improvement over that of Lazebnik et al. (2005), and reported classification rates of 95.3% and 98.7% on CUReT and UIUC respectively. They recommended that practical texture recognition should seek to incorporate multiple types of complementary features, but with local invariance properties not exceeding those absolutely required for a given application. Other local region detectors have also been used for texture description, such as the Scale Descriptors which measure the scales of salient textons (Kadir and Brady 2002).

3.2.2 Dense Texture Descriptors

The number of features derived from a sparse set of interesting points is much smaller than the total number of image pixels, resulting a compact feature space. However, the sparse approach can be inappropriate for many texture classification tasks:

- Interest point detectors typically produce a sparse output and could miss important texture elements.

[1] Originally, SIFT is comprised of a detector and descriptor, but which are used in isolation now; in this survey, if not specified, SIFT refers to the descriptor, a common practice in the community.

Table 1 A summary of components in the BoW representation pipeline, as was sketched in Fig. 5

Step	Approach	Highlights
Local Texture Descriptors (Sect. 3.2)	Sparse Descriptors	
	(Harris + Laplacian) (RIFT + SPIN) (Lazebnik et al. 2005)	Keypoint detectors plus novel descriptors SPIN and RIFT
	(Harris + Laplacian) (RIFT + SPIN + SIFT) (Zhang et al. 2007)	A comprehensive evaluation of multiple keypoint detectors, feature descriptors, and classifier kernels
	Dense Descriptors	
	Gabor Wavelets	Joint optimum resolution in time and frequency; Multiscale and multiorientation analysis
	LMfilters (Leung and Malik 2001)	First to propose Bag of Texton (BoT) model (i.e. the BoW model)
	Schmid Filters	Gabor like filters; Rotation invariant
	MR8 (Varma and Zisserman 2005)	Rotationally invariant filters and low-dimensional filter response space
	Patch Intensity (Varma and Zisserman 2009)	Challenge the dominant role of filter descriptors and propose image raw intensity feature
	LBP (Ojala et al. 2002b)	Fast binary features with gray scale invariance; Predefined codebook
	Random Projection (Liu and Fieguth 2012)	First to introduce compressive sensing and random projection into texture classification
	Sorted Random Projection (Liu et al. 2011a)	Efficient and effective approach for random projection to achieve rotation invariance
	Basic Image Features (BIFs) (Crosier and Griffin 2010)	Introduce BIFs of Griffin and Lillholm into texture classification; Predefined codebook
	Weber Local Descriptor (WLD) (Crosier and Griffin 2010)	A descriptor based on Weber's Law
	Fractal Based Descriptors	
	MultiFractal Spectrum (Xu et al. 2009b)	Invariant under the bi-Lipschitz mapping
Codebook Generation (Sect. 3.3)	Predefined (Crosier and Griffin 2010; Ojala et al. 2002b)	No codebook learning step; Computationally efficient
	*k*means clustering (Csurka et al. 2004; Leung and Malik 2001)	Most commonly used method; Cannot capture overlapping distributions in the feature space
	GMM modeling (Cimpoi et al. 2016; Perronnin et al. 2010; Sharma and Jurie 2016)	Considers both cluster centers and covariances which describe the spreads of clusters
	Sparse Coding based learning (Peyré 2009; Skretting and Husøy 2006)	Sparse representation based; Minimize reconstruction error of data; Computationally expensive
Feature Encoding (Sect. 3.4)	Voting Based Methods	Require a large codebook (usually learned by *k*means); Usually combine with nonlinear SVM
	Hard Voting (Leung and Malik 2001; Varma and Zisserman 2005)	Quantize each feature to nearest codeword; Fast to compute; Codes are sparse and high dimensional
	Soft Voting (Ahonen and Pietikäinen 2007; Ren et al. 2013; Van Gemert et al. 2008)	Assigns each feature to multiple codewords; Does not minimize reconstruction error
	Fisher Vector (FV) Based Methods	Require a small codebook; Very high dimension; Combines with efficient linear SVM
	FV (Perronnin and Dance 2007)	GMM-based; Encodes higher order statistics; Efficient to compute
	Improved FV (IFV) (Cimpoi et al. 2014; Perronnin et al. 2010; Sharma and Jurie 2016)	Uses signed square rooting and L_2 normalization; State of the art performance in texture classification
	VLAD (Jegou et al. 2012; Cimpoi et al. 2014)	A simplified version of FV

Table 1 continued

Step	Approach	Highlights
	Reconstruction Based Methods	Enforce sparse representation; Explores the manifold structure of data; Minimize reconstruction error
	Sparse Coding (Peyré 2009; Skretting and Husøy 2006; Yang et al. 2009)	Leverage that fact that natural images are sparse; Optimization is computationally expensive
	Local constraint Linear Coding (LLC) (Cimpoi et al. 2014; Wang et al. 2010)	Local smooth sparsity; Fast computation through approximated LLC
Feature Pooling (Sect. 3.5)	Average Pooling	The most widely used pooling scheme in texture representation
	Max Pooling	Usually used in combination with sparse coding and LLC
	Spatial Pyramid Pooling (SPM)	Preserving more spatial information; Higher feature dimensionality
Classifier (Sect. 3.5)	Nearest Neighbor Classifier (NNC) (Liu and Fieguth 2012; Varma and Zisserman 2005)	Simple and elegant nonparametric classifier; Popular in texture classification
	Kernel SVM (Zhang et al. 2007)	Usually in combination with Chi Square for BoW based representation
	Linear SVM (Cimpoi et al. 2016)	Suitable for high-dimensional feature representation like FV and VLAD

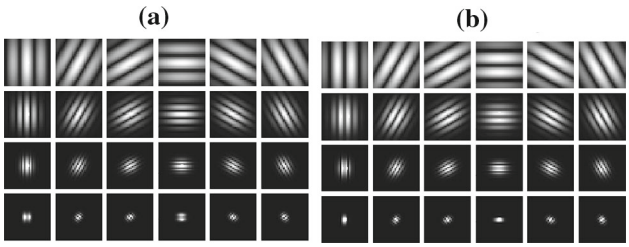


Fig. 6 Illustration of the Gabor wavelets used in Manjunath and Ma (1996). **a** Real part, **b** Imaginary part

- A sparse output in a small image might not produce sufficient regions for robust statistical characterization.
- There are issues regarding the repeatability of the detectors, the stability of the selected regions and the instability of orientation estimation (Mikolajczyk et al. 2005).

As a result, extracting local texture features *densely* at each pixel is the more popular representation, the subject of the following discussion.

(1) Gabor Filters are one of the most popular texture descriptors, motivated by their relation to models of early visual systems of mammals as well as their joint optimum resolution in time and frequency (Jain and Farrokhnia 1991; Lee 1996; Manjunath and Ma 1996). As illustrated in Fig. 6, Gabor filters can be considered as orientation and scale tunable edge and bar detectors. The Gabor wavelets are generated by appropriate rotations and dilations from the following product of an elliptical Gaussian and a complex plane wave:

$$\phi(x, y) = \frac{1}{2\pi\sigma_x\sigma_y}\exp\left[-\left(\frac{x^2}{2\sigma_x^2}+\frac{y^2}{2\sigma_y^2}\right)\right]\exp(j2\pi\omega),$$

whose Fourier transform is

$$\hat{\phi}(x, y) = \exp\left[-\left(\frac{(u-\omega)^2}{2\sigma_u^2}+\frac{v^2}{2\sigma_v^2}\right)\right],$$

where ω is the radial center frequency of the filter in the frequency domain, σ_x and σ_y are the standard deviations of the elliptical Gaussian along x and y.

Thus, a Gabor filter bank is defined by its parameters including frequencies, orientations and the parameters of the Gaussian envelope. In the literature, different parameter settings have been suggested, and filter banks created by these parameter settings work well in general. Details for the derivation of Gabor wavelets and parameter selection can be found in Lee (1996), Manjunath and Ma (1996), Petrou and Sevilla (2006). Invariant Gabor representations can be accessed in Han and Ma (2007). According to the experimental study in Kandaswamy et al. (2011) and Zhang et al. (2007), Gabor features (Manjunath and Ma 1996) fail to meet the expected level of performance in the presence of rotation, affine and scale variations. However, Gabor filters encode structural features from multiple orientations and over a broader range of scales. It has been shown (Kandaswamy et al. 2011) that for large datasets, under varying illumination conditions, Gabor filters can serve as a preprocessing method and combine with LBP (Ojala et al. 2002b) to obtain texture features with reasonable robustness (Pietikäinen et al. 2011; Zhang et al. 2005).

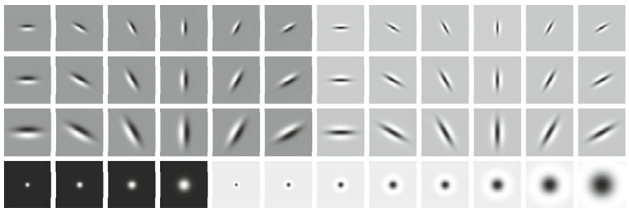


Fig. 7 The LMfilter bank has a mix of edge, bar and spot filters at multiple scales and orientations. It has a total of 48 filters: 2 Gaussian derivative filters at 6 orientations and 3 scales, 8 Laplacian of Gaussian filters and 4 Gaussian filters

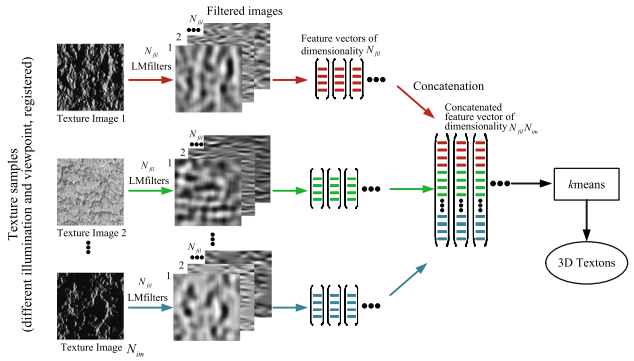


Fig. 8 Illustration of the process of 3D texton dictionary learning proposed by Leung and Malik (2001). Each image at different lighting and viewing directions is filtered using the filter bank illustrated in Fig. 7. The response vectors are concatenated together to form data vectors of length $N_{fil}N_{im}$. These data vectors are clustered using the *k*means algorithm to obtain the 3D textons

(2) Filters by Leung and Malik (LM Filters) Researchers (Leung and Malik 2001; Malik et al. 1999) pioneered the problem of classifying textures under varying viewpoint and illumination. The LM filters used for local texture feature extraction are illustrated in Fig. 7. In particular, they marked a milestone by giving an operational definition of textons: the cluster centers of the filter response vectors. Their work has been widely followed by other researchers (Csurka et al. 2004; Lazebnik et al. 2005; Shotton et al. 2009; Sivic and Zisserman 2003; Varma and Zisserman 2005, 2009). To handle 3D effects caused by imaging, they proposed 3D textons which were cluster centers of filter responses over a stack of images with representative viewpoints and lighting, as illustrated in Fig. 8. In their texture classification algorithm, 20 images of each texture were geometrically registered and transformed into 48D local features with the LM Filters. Then the 48D filter response vectors of 20 selected images of the same pixel were concatenated to obtain a 960D feature vector as the local texture representation, subsequently input into a BoW pipeline for texture classification. A downside of the method is that it is not suitable for classifying a single texture image under *unknown* imaging conditions, which usually arises in practical applications.

(3) The Schmid Filters (S Filters) (Schmid 2001) consist of 13 rotationally invariant Gabor-like filters of the form

$$\phi(x, y) = \exp\left[-\left(\frac{x^2+y^2}{2\sigma^2}\right)\right] cos\left(\frac{\pi\beta\sqrt{x^2+y^2}}{\sigma}\right),$$

where β is the number of cycles of the harmonic function within the Gaussian envelope of the filter. The filters are shown in Fig. 9; as can be seen, all of the filters have rotational symmetry. The rotation-invariant S Filters were shown to outperform the rotation-variant LM Filters in classifying the CUReT textures (Varma and Zisserman 2005), indicating that rotational invariance is necessary in practical applications.

(4) Maximum Response (MR8) Filters of Varma and Zisserman (2005) consist of 38 root filters but only 8 filter responses. The filter bank contains filters at multiple orientations but their outputs are pooled by recording only the maximum filter response across all orientations, in order to achieve rotation invariance. The root filters are a subset of the LM Filters (Leung and Malik 2001) of Fig. 7, retaining the two rotational symmetry filters, the edge filter, and the bar filter at 3 scales and 6 orientations. Recording only the maximum response across orientations reduces the number of responses from 38 to 8 (3 scales for 2 anisotropic filters, plus 2 isotropic), resulting the so called MR8 filter bank.

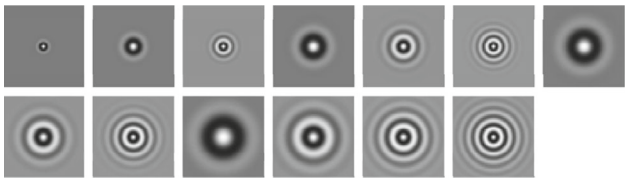


Fig. 9 Illustration of the rotationally invariant Gabor-like Schmid filters used in Schmid (2001). The parameter (σ, β) pair takes values (2,1), (4,1), (4,2), (6,1), (6,2), (6,3), (8,1), (8,2), (8,3), (10,1), (10,2), (10,3) and (10,4)

Realizing the shortcomings of Leung and Malik's method (2001), Varma and Zisserman (2005) attempted to improve the classification of a single texture sample image under unknown imaging conditions, bypassing the registration step, instead learning 2D textons by aggregating filter responses over different images. Experimental results (Varma and Zisserman 2005) showed that MR8 outperformed the LM Filters and S Filters, indicating that detecting better features and clustering in a lower dimensional feature space can be advantageous. The best results for MR8 are 97.4% obtained with a dictionary of 2440 textons and a Nearest Neighbor Classifier (NNC) (Varma and Zisserman 2005). Later, Hayman et al. (2004) showed that SVM could further enhance the texture classification performance of MR8 features, giving a 98.5% classification rate for the same setup used for texton representation.

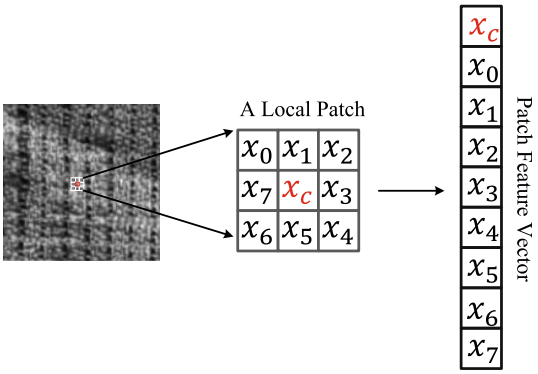


Fig. 10 Illustration for the Patch Descriptor proposed in Varma and Zisserman (2009): the raw intensity vector is used directly as the local representation

(5) Patch Descriptors of Varma and Zisserman (2009) challenged the dominant role of the filter banks (Mellor et al. 2008; Randen and Husoy 1999) in texture analysis, and instead developed a simple Patch Descriptor, keeping the raw pixel intensities of a square neighborhood to form a feature vector, as illustrated in Fig. 10. By replacing the filter responses such as LM Filters (Randen and Husoy 1999), S Filters (Schmid 2001) and MR8 (Varma and Zisserman 2005) with the Patch Descriptor in texture classification, Varma and Zisserman (2009) observed very good classification performance using extremely compact neighborhoods (3×3), and that for any fixed size of neighborhood the Patch Descriptor leads to superior classification compared to filter banks with the same support.

Two variants of the Patch Descriptor, the Neighborhood Descriptor and the MRF Descriptor, were developed. For the Neighborhood Descriptor, the central pixel is discarded and only the neighborhood vector is used for texton representation. Instead of ignoring the central pixel, the MRF Descriptor explicitly models the joint distribution of the central pixels and its neighbors. The best result 98.0% is given by the MRF Descriptor using a 7×7 neighborhood with 2440 textons and 90 bins and a NNC classifier. Note that the dimensionality of this MRF representation is very high: 2440×90. A clear limitation of the Patch, Neighborhood and MRF Descriptors is sensitivity to nearly any change (brightness, rotation, affine etc.). Varma and Zisserman (2009) adopted the method of finding the dominant orientation of a patch and measuring the neighborhood relative to this orientation to achieve rotation invariance, and reported a 97.8% classification rate on the UIUC dataset. It is worth mentioning that finding the dominant orientation for each patch is computationally expensive.

(6) Random Projection (RP) and Sorted Random Projection (SRP) features of Liu and Fieguth (2012) were inspired by theories of sparse representation and compressed sensing (Candes and Tao 2006; Donoho 2006). Taking advantage of the sparse nature of textured images, a small set of random features is extracted from local image patches by projecting the local patch feature vectors to a lower dimensional feature subspace. The random projection is a fixed, distance-preserving embedding capable of alleviating the curse of dimensionality (Baraniuk et al. 2008; Giryes et al. 2016). The random features are embedded into BoW to perform texture classification. It has been shown that the performance of RP features is superior to that of the Patch Descriptor with equivalent neighborhoods (Liu and Fieguth 2012); a clear indication that the RP matrix preserves the salient information contained in the local patch and that performing classification in a lower feature space is advantageous. The best result 98.5% is achieved using a 17×17 neighborhood with 2440 textons and a NNC classifier.

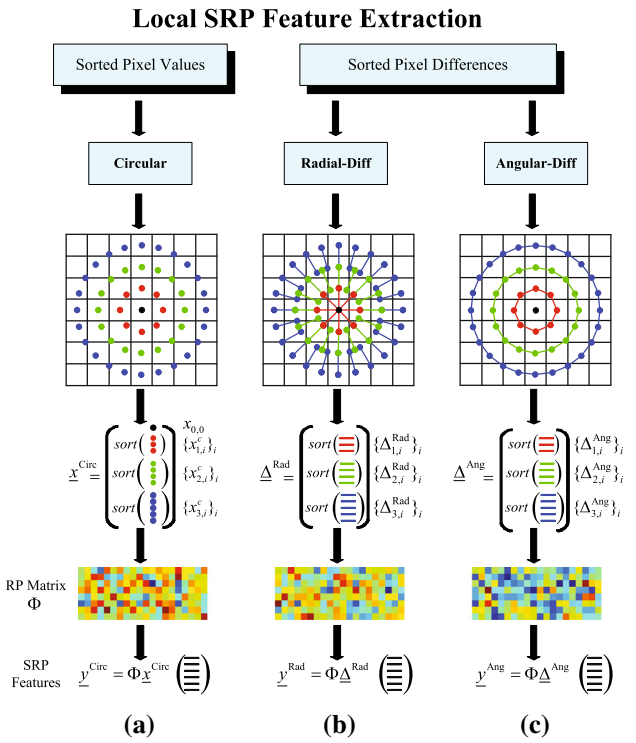


Fig. 11 An illustration of SRP descriptor: extracting SRP features on an example local image patch of size 7×7. **a** Sorting pixel intensities; **b**, **c** sorting pixel differences

Like the Patch Descriptors, the RP features remain sensitive to image rotation. To further improve robustness, Liu et al. (2011a, 2012) proposed sorting the RP features, as illustrated in Fig. 11, whereby rings of pixel values are sorted, without any reference orientation, ensuring rotation invariance. Two kinds of local features are used, one based on raw intensities and the other on gradients (radial differences and angular differences). Random functions of the sorted local features are taken to obtain SRP features. It was shown that SRP outperformed RP significantly for robust texture classification (Liu et al. 2011a, 2012), producing state of the art

classification results on CUReT (99.4%) KTHTIPS (99.3%), and UMD (99.3%) with a SVM classifier (Liu et al. 2011a, 2015).

(7) Local Binary Patterns of Ojala et al. (1996) marked the beginning of the LBP methodology, followed by the simpler rotation invariant version of Pietikäinen et al. (2000), and later "uniform" patterns to reduce feature dimensionality (Ojala et al. 2002b).

Texture representation generally requires the analysis of patterns in local pixel neighborhoods, which are comprehensively described by their joint distribution. However, stable estimation of joint distributions is often infeasible, even for small neighborhoods, because of the combinatorics of joint distributions. Considering the joint distribution:

$$g(x_c, x_0, \ldots, x_{p-1}) \tag{1}$$

of center pixel x_c and $\{x_n\}_{n=0}^{p-1}$, p equally spaced pixels on a circle of radius r, Ojala et al. (2002b) argued that much of the information in this joint distribution is conveyed by the joint distribution of differences:

$$g(x_0 - x_c, x_1 - x_c, \ldots, x_{p-1} - x_c). \tag{2}$$

The size of the joint histogram was greatly minimized by keeping only the *sign* of each difference, as illustrated in Fig. 12.

A certain degree of rotation invariance is achieved by cyclic shifts of the LBPs, i.e., grouping together those LBPs that are actually rotated versions of the same underlying pattern. Since the dimensionality of the representation (which grows exponentially with p) is still high, Ojala et al. (2002b) introduced a uniformity measure to identify $p(p-1)+2$ uniform LBPs and classified all remaining nonuniform LBPs under a single group. By changing parameters p and r, we can derive LBP for any quantization of the angular space and for any spatial resolution, such that multiscale analysis can be accomplished by combining multiple operators of varying r. The most prominent advantages of LBP are its invariance to monotonic gray scale change, very low computational complexity, and ease of implementation.

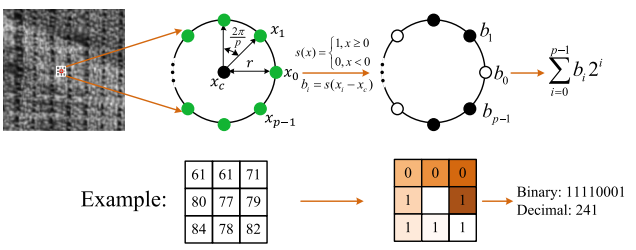


Fig. 12 A circular neighborhood used to derive an LBP code: a central pixel x_c and its p circularly and evenly spaced neighbors on a circle of radius r

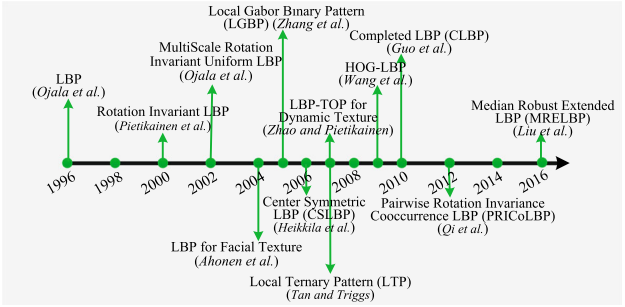


Fig. 13 LBP and its representative variants (see text for discussion)

Since (Ojala et al. 2002b), LBP started to receive increasing attention in computer vision and pattern recognition, especially texture and facial analysis, with the LBP milestones presented in Fig. 13. As Gabor filters and LBP provide complementary information (LBP captures small and fine details, Gabor filters encode appearance information over a broader range of scales), Zhang et al. (2005) proposed Local Gabor Binary Pattern (LGBP) by extracting LBP features from images filtered by Gabor filters of different scales and orientations, to enhance the representation power, followed by subsequent Gabor-LBP approaches (Huang et al. 2011; Liu et al. 2017; Pietikäinen et al. 2011). Additional important LBP variants include LBP-TOP, proposed by Zhao and Pietikäinen (2007), a milestone in using LBP for dynamic texture analysis; the Local Ternary Patterns (LTP) of Tan and Triggs (2007), introducing a pair of thresholds and a split coding scheme which allows for encoding pixel similarity; the Local Phase Quantization (LPQ) by Ojansivu and Heikkilä (2008), Ojansivu et al. (2008) quantizing the Fourier transform phase in local neighborhoods which is, by design, tolerant to most common types of image blurs; the Completed LBP (CLBP) of Guo et al. (2010), encoding not only the signs but also the magnitudes of local differences; and the Median Robust Extended LBP (MRELBP) of Liu et al. (2016b) which enjoys high distinctiveness, low computational complexity, and strong robustness to image rotation and noise.

LBP has also led to compact and efficient binary feature descriptors designed for image matching, with noticeable ones including Binary Robust Independent Elementary Features (BRIEF) (Calonder et al. 2012), Oriented FAST and Rotated BRIEF (ORB) (Rublee et al. 2011), Binary Robust Invariant Scalable Keypoints (BRISK) (Leutenegger et al. 2011) and Fast Retina Keypoint (FREAK) (Alahi et al. 2012). These binary descriptors provide a comparable matching performance with the widely used region descriptors such as SIFT (Lowe 2004) and SURF (Bay et al. 2006), but are fast to compute and have significantly lower memory requirements, especially suitable for applications on resource constrained devices.

In summary, for large datasets with rotation variations and no significant illumination related variations, LBP (Ojala

et al. 2002b) could serve as an effective and efficient approach for texture classification. However, in the presence of significant illumination variations, significant affine transformations, or noise corruption, LBP fails to meet the expected level of performance. MRELBP (Liu et al. 2016b), a recent LBP variant, has been demonstrated to outperform LBP significantly, with near perfect classification performance on two small benchmark datasets (Outex_TC10 100% and Outex_TC12 99.8%) (Liu et al. 2016b), and which obtained the best overall performance in a recent experimental survey (Liu et al. 2017) evaluating robustness in multiple classification challenges. In general, LBP-based features work well in situations when limited training data are available; learning based approaches like MR8, Patch Descriptors and DCNN based representations, which require large amount of training samples, are significantly outperformed by LBP based ones.

After over 20 years of developments, LBP is no longer just a simple texture operator, but has laid the foundation for a direction of research dealing with local image and video descriptors. A large number of LBP variants have been proposed to improve its robustness and to increase its discriminative power and applicability to different types of problems, and interested readers are referred to excellent surveys (Huang et al. 2011; Liu et al. 2017; Pietikäinen et al. 2011). Recently, although CNN based methods are beginning to dominate, LBP research remains active, as evidenced by significant recent work (Guo et al. 2016; Sulc and Matas 2014; Ryu et al. 2015; Levi and Hassner 2015; Lu et al. 2018; Xu et al. 2017; Zhai et al. 2015; Ding et al. 2016).

(8) Basic Image Features (BIF) approach (Crosier and Griffin 2010) is similar to LBP (Ojala et al. 2002b), in that it is based upon a predefined codebook rather than one learned from training. It therefore shares the advantages of LBP over methods based on codebook learning with clustering. In contrast with LBP, BIF probes an image locally using Gaussian derivative filters (Griffin and Lillholm 2010; Griffin et al. 2009) whereas LBP computes the differences between a pixel and its neighbors. Derivative of Gaussians (DtG), consisting of first and second order derivatives of the Gaussian filter, can effectively detect the local basic and symmetry structure of an image, and allows achieving exact rotation invariance (Freeman and Adelson 1991). BIF feature extraction is summarized in Fig. 14: each pixel in the image is filtered by the DtG filters, and then labeled as the maximizing class. A simple six dimensional BIF histogram can be used as a global texture representation, however the histogram over these six categories produces too coarse a representation, therefore others (e.g., Crosier and Griffin 2010) have performed multiscale analysis and calculated joint histograms over multiple scales. Multiscale BIF features achieved very good classification performance on CUReT (98.6%), UIUC (98.8%) and

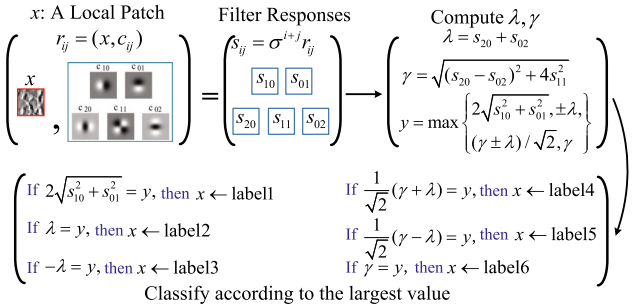
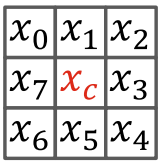


Fig. 14 Illustration of the calculation of BIF features

Fig. 15 First order square symmetric neighborhood for WLD computation



KTHTIPS (98.5%) (Crosier and Griffin 2010), with a NNC classifier.

(9) Weber Law Descriptor (WLD) (Chen et al. 2010) is based on the fact that human perception of a pattern depends not only on the change of a stimulus but also on the original intensity of the stimulus. The WLD consists of two components: differential excitation and orientation. For a small patch of size 3×3, shown in Fig. 15, the differential excitation is the relative intensity ratio

$$\xi(x_c) = \arctan\left(\frac{\sum_{i=0}^{7}(x_i - x_c)}{x_c}\right)$$

and the orientation component is derived from the local gradient orientation

$$\theta(x_c) = \arctan\frac{x_7 - x_3}{x_5 - x_1}.$$

Both ξ and θ are quantified into a 2D histogram, offering a global representation. Clearly the use of multiple neighborhood sizes supports a multiscale generalization. Though computationally efficient, WLD features fail to meet the expected level of performance for texture recognition.

3.2.3 Fractal Based Descriptors

Fractal Based Descriptors present a mathematically well founded alternative to dealing with scale (Mandelbrot and Pignoni 1983), however they have not become popular as texture features due to their lack of discriminative power (Varma and Garg 2007). Recently, inspired by the BoW approach, researchers revisited the fractal method and proposed the MultiFractal Spectrum (MFS) method (Xu et al. 2009a, b, 2010), invariant to viewpoint changes, nonrigid deformations and local affine illumination changes.

The basic MFS method was proposed in Xu et al. (2009b), where MFS was first defined for simple image features, such as intensity, gradient and Laplacian of Gaussian (LoG). A texture image is first transformed into n feature maps such as intensity, gradient or LoG filter features. Each map is clustered into k clusters (i.e. k codewords) via $k$means. Then, a codeword label map is obtained and is decomposed into k binary feature maps: those pixels assigned to codeword i are labeled with 1 and the remainder as 0. For each binary feature map, the box counting algorithm (Xu et al. 2010) is used to estimate a fractal dimension feature. Thus, a total of k fractal dimension features are computed for each feature map, forming a kD feature vector (referred to as a fractal spectrum) as the global representation of the image. Finally, for the n different feature maps, n fractal spectrum feature vectors are concatenated as the MFS feature. The MFS representation demonstrated invariance to a number of geometrical changes such as viewpoint changes, nonrigid surface changes and reasonable robustness to illumination changes. However, since it is based on simple features (intensities and gradients) and has very low dimension, it has limited discriminability, and gives classification rates 92.3% and 93.9% on datasets UIUC and UMD respectively.

Later MFS was improved by generalizing the simple image intensity and gradient features with SIFT (Xu et al. 2009a), wavelets (Xu et al. 2010), and LBP (Quan et al. 2014). For instance, the Wavelet based MFS (WMFS) features archived significantly improved classification performance on UIUC (98.6%) and UMD (98.7%). The downside of the MFS approach is that it requires high resolution images to obtain sufficiently stable features.

3.3 Codebook Generation

Texture characterization requires the analysis of spatially repeating patterns, which suffice to characterize textures and the pursuit of which has had important implications in a series of practical problems, such as dimensionality reduction, variable decoupling, and biological modelling (Olshausen and Field 1997; Zhu et al. 2005). The extracted set of local texture features is versatile, and yet overly redundant (Leung and Malik 2001). It can therefore be expected that a set of prototype features (i.e. codewords or textons) must exist which can be used to create global representations of textures in natural images (Leung and Malik 2001; Okazawa et al. 2015; Zhu et al. 2005), in a similar way as in speech and language (such as words, phrases and sentences).

There exist a variety of methods for codebook generation. Certain approaches, such as LBP (Ojala et al. 2002b) and BIF (Crosier and Griffin 2010), which we have already discussed, use predefined codebooks, therefore entirely bypassing the codebook learning step.

For approaches requiring a learned codebook, $k$means clustering (Lazebnik et al. 2005; Leung and Malik 2001; Liu and Fieguth 2012; Varma and Zisserman 2009; Zhang et al. 2007) and Gaussian Mixture Models (GMM) (Cimpoi et al. 2014, 2016; Lategahn et al. 2010; Jegou et al. 2012; Perronnin et al. 2010; Sharma and Jurie 2016) are the most popular and successful methods. GMM modeling considers both cluster centers and covariances, which describe the location and spread/shape of clusters, whereas $k$means clustering cannot capture overlapping distributions in the feature space as it considers only distances to cluster centers, although generalizations to $k$means with multiple prototypes per cluster can allow this limitation to be relaxed. The GMM and $k$means methods learn a codebook in an unsupervised manner, but some recent approaches focus on building more discriminative ones (Yang et al. 2008; Winn et al. 2005).

In addition, another significant research thread is reconstruction based codebook learning (Aharon et al. 2006; Peyré 2009; Skretting and Husøy 2006; Wang et al. 2010), under the assumption that natural images admit a sparse decomposition in some redundant basis (i.e., dictionary or codebook). These methods focus on learning nonparametric redundant dictionaries that facilitate a sparse representation of the data and minimize the reconstruction error of the data. Because discrimination is the primary goal of texture classification, researchers have proposed to construct discriminative dictionaries that explicitly incorporate category specific information (Mairal et al. 2008, 2009).

Since the codebook is used as the basis for encoding feature vectors, codebook generation is often interleaved with feature encoding, described next.

3.4 Feature Encoding

As illustrated in Fig. 4, a given image is transformed into a pool of local texture features, from which a global image representation is derived by feature encoding with the generated codebook. In the field of texture classification, we group commonly-used encoding strategies into three major categories:

- Voting based (Leung and Malik 2001; Varma and Zisserman 2005; Van Gemert et al. 2008; Van Gemert et al. 2010),
- Fisher Vector based (Jegou et al. 2012; Cimpoi et al. 2016; Perronnin et al. 2010; Sanchez et al. 2013), and
- Reconstruction based (Mairal et al. 2008, 2009; Olshausen and Field 1996; Peyré 2009; Wang et al. 2010).

Comprehensive comparisons of encoding methods in image classification can be found in Chatfield et al. (2011), Cimpoi et al. (2014), Huang et al. (2014).

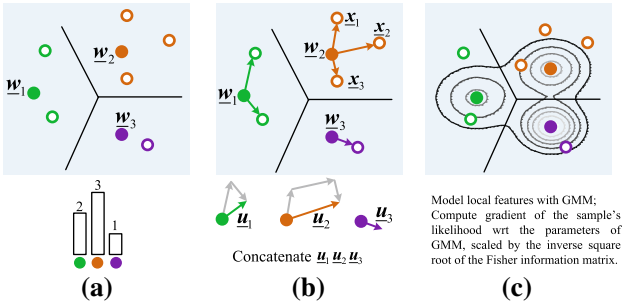


Fig. 16 Contrasting the ideas of BoW, VLAD and FV. **a** BoW: Counting the number of local features assigned to each codeword. It encodes the zero order statistics of the distribution of local descriptors. **b** VLAD: accumulating the differences of local features assigned to each codeword. **c** FV: The Fisher vector extends the BOW by encoding higher order statistics (first and second order), retaining information about the fitting error of the best fit

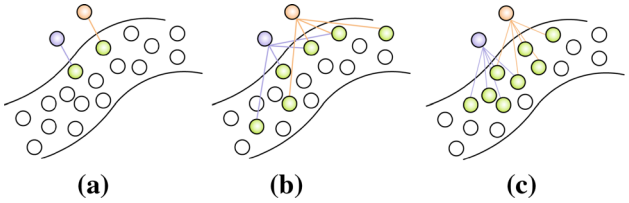


Fig. 17 Contrasting the ideas of hard voting, sparse coding, and LLC. **a** Encoding with hard voting, **b** encoding with sparse coding, **c** encoding with LLC

Voting based methods The most intuitive way to quantize a local feature is to assign it to its nearest codeword in the codebook, also referred to as hard voting (Leung and Malik 2001; Varma and Zisserman 2005). A histogram of the quantized local descriptors can be computed by counting the number of local features assigned to each codeword; this histogram constitutes the baseline BoW representation (as illustrated in Fig. 16a) upon which other methods can improve. Formally, it starts by learning a codebook $\{\boldsymbol{w}_i\}_{i=1}^{K}$, usually by *k*means clustering. Given a set of local texture descriptors $\{\boldsymbol{x}_i\}_{i=1}^{N}$ extracted from an image, the encoding representation of some descriptor $\boldsymbol{x}$ via hard voting is

$$v(i) = \begin{cases} 1, \text{ if } i = \text{argmin}_j(\|\boldsymbol{x} - \boldsymbol{w}_j\|) \\ 0, \text{ otherwise.} \end{cases} \tag{3}$$

The histogram of the set of local descriptors is to aggregate all encoding vectors $\{\boldsymbol{v}_i\}_{i=1}^{N}$ via sum pooling. Hard voting overlooks codeword uncertainty, and may label image features by nonrepresentative codewords. In an improvement to this hard voting scheme, soft voting (Ahonen and Pietikäinen 2007; Ren et al. 2013; Ylioinas et al. 2013; Van Gemert et al. 2008; Van Gemert et al. 2010) employs several nearest codewords to encode each local feature in a soft manner, such that the weight of each assigned codeword is an inverse function of the distance from the feature, for some kernel definition of distance. Voting based methods yield a histogram representation of dimensionality K, the number of bins in the histogram.

Fisher Vector based methods By counting the number of occurrences of codewords, the standard BoW histogram representation encodes the zeroth-order statistics of the distribution of descriptors, which is only a rough approximation of the probability density distribution of the local features. The Fisher vector extends the histogram approach by encoding additional information about the distribution of the local descriptors. Based on the original FV encoding (Perronnin and Dance 2007), improved versions were proposed (Cinbis et al. 2016; Perronnin et al. 2010) such as the Improved FV (IFV) (Perronnin et al. 2010) and VLAD (Jegou et al. 2012). We briefly describe IFV (Perronnin et al. 2010) here, since to the best of our knowledge it achieves the best performance in texture classification (Cimpoi et al. 2014, 2015, 2016; Sharma and Jurie 2016). Theory and practical issues regarding FV encoding can be found in Sanchez et al. (2013).

IFV encoding learns a soft codebook with GMM, as shown in Fig. 16c. An IFV encoding of a local feature is computed by assigning it to each codeword, in turn, and computing the gradient of the soft assignment with respect to the GMM parameters.[2] The IFV encoding dimensionality is $2DK$, where D is the dimensionality of the feature space and K is the number of Gaussian mixtures. BoW can be considered a special case of FV in the case where the gradient computation is restricted to the mixture weight parameters of the GMM. Unlike BoW, which requires a large codebook size, FV can be computed from a much smaller codebook (typically 64 or 256) and therefore at a lower computational cost at the codebook learning step. On the other hand, the resulting dimension of the FV encoding vector (e.g. tens of thousands) is usually significantly higher than BoW (thousands), which makes it unsuitable for nonlinear classifiers, however it offers good performance even with simple linear classifiers.

The VLAD encoding scheme proposed by Jegou et al. (2012) can be thought of as a simplified version of FV, in that it typically uses *k*means, rather than GMM, and records only first-order statistics rather than second order. In particular, it records the residuals (the difference between the local features and the codewords), as shown in Fig. 16b.

Reconstruction based methods Reconstruction based methods aim to obtain an information-preserving encoding vector that allows for the reconstruction of a local feature with a small number of codewords. Typical methods include sparse coding and Local constraint Linear Coding (LLC), which are contrasted in Fig. 17. Sparse coding was initially proposed (Olshausen and Field 1996) to model natural image

[2] The derivative to weights, which is considered to make little contribution to the performance, is removed in IFK (Perronnin et al. 2010).

statistics, then to texture classification (Dahl and Larsen 2011; Mairal et al. 2008, 2009; Peyré 2009; Skretting and Husøy 2006) and later to other problems such as image classification (Yang et al. 2009) and face recognition (Wright et al. 2009).

In sparse coding, a local feature $\boldsymbol{x}$ can be well approximated by a sparse decomposition $\boldsymbol{x} \approx \mathbf{W}\boldsymbol{v}$ over the learned codebook $\mathbf{W} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots \boldsymbol{w}_K]$, by leveraging the sparse nature of the underlying image (Olshausen and Field 1996). A sparse encoding can be solved as

$$\text{argmin}_{\boldsymbol{v}} \|\boldsymbol{x} - \mathbf{W}\boldsymbol{v}\|_2^2 \quad s.t. \quad \|\boldsymbol{v}\|_0 \leq s. \tag{4}$$

where s is a small integer denoting the sparsity level, limiting the number of nonzero entries in $\boldsymbol{v}$, measured as $\|\boldsymbol{v}\|_0$. Learning a redundant codebook that facilitate a sparse representation of the local features is important in sparse coding (Aharon et al. 2006). Methods in Mairal et al. (2008, 2009), Peyré (2009), Skretting and Husøy (2006) are based on learning C class-specific codebooks, one for each texture class and approximating each local feature using a constant sparsity s. The C different codebooks provides C different reconstruction errors, which can then be used as classification features. In Peyré (2009) and Skretting and Husøy (2006), the class specific codebooks were optimized for reconstruction, but significant improvements have been shown by optimizing for discriminative power instead (Dahl and Larsen 2011; Mairal et al. 2008, 2009), an approach which is, however, associated with high computational cost, especially when the number of texture classes C is large.

Locality constrained linear coding (LLC) (Wang et al. 2010) projects each local descriptor $\boldsymbol{x}$ down to the *local* linear subspace spanned by q codewords in the codebook of size K closest to it (in Euclidean distance), resulting in a K dimensional encoding vector whose entries are all zero except for the indices of the q codewords closest to $\boldsymbol{x}$. The projection of $\boldsymbol{x}$ down to the span of its q closest codewords is solved via

$$\text{argmin}_{\boldsymbol{v}} \|\boldsymbol{x} - \mathbf{W}\boldsymbol{v}\|_2^2 + \lambda \sum_{k=1}^{K} \left(\boldsymbol{v}(i) exp \frac{\|\boldsymbol{x} - \boldsymbol{w}_i\|_2}{\sigma} \right)^2$$

$$s.t. \sum_{k=1}^{K} \boldsymbol{v}(i) = 1,$$

where λ is a small regularization constant and σ adjusts the weight decay speed.

In summary, reconstruction based coding has received significant attention since sparse coding was applied for visual classification (Mairal et al. 2008, 2009; Peyré 2009; Skretting and Husøy 2006; Wang et al. 2010). A theoretical study for the success of sparse coding over vector quantization can be found in Coates and Ng (2011).

3.5 Feature Pooling and Classification

The goal of feature pooling (Boureau et al. 2010) is to integrate or combine the coded feature vectors $\{\boldsymbol{v}_i\}_i, \boldsymbol{v}_i \in \mathbb{R}^d$ of a given image into a final compact global representation $\boldsymbol{y}_i$ which is more robust to image transformations and noise. Commonly used pooling methods include sum pooling, average pooling and max pooling (Leung and Malik 2001; Varma and Zisserman 2009; Wang et al. 2010). Boureau et al. (2010) presented a theoretical analysis of average pooling and max pooling, and showed that max pooling may be well suited to sparse features. The authors also proposed softer max pooling methods by using a smoother estimate of the expected max-pooled feature and demonstrated improved performance. Another noticeable pooling method is the mix-order max pooling method which considers the information of visual word occurrence frequency (Liu et al. 2011b).

Specifically, let $\mathbf{V} = [\boldsymbol{v}_1, ..., \boldsymbol{v}_N] \in \mathbb{R}^{d \times N}$ denote the coded features from N locations. For $\boldsymbol{u}$ denoting a row of $\mathbf{V}$, $\boldsymbol{u}$ is reduced to a single scalar by some operation (sum, average, max), reducing $\mathbf{V}$ to a d-dimensional feature vector. Realizing that pooling over the entire image disregards all information regarding spatial dependencies, Lazebnik et al. (2006) proposed a simple Spatial Pyramid Pooling (SPM) scheme by partitioning the image into increasingly fine subregions and computing histograms of local features found inside each subregion via average or max pooling. The final global representation is a concatenation of all histograms extracted from subregions, resulting in a higher dimensional representation that preserves more spatial information (Timofte and Van Gool 2012).

Given a pooled feature, a given texture sample can be classified. Many classification approaches are possible (Jain et al. 2000; Webb and Copsey 2011), although Nearest Neighbor Classifier (NNC) and Support Vector Machine (SVM) are the most widely-used classifiers for the BoW representation. Different distance measures may be used, such as the EMD distance (Lazebnik et al. 2005; Zhang et al. 2007), KL divergence and the widely-used Chi Square distance (Liu and Fieguth 2012; Varma and Zisserman 2009). For high dimensional BoW features, as with SPM features and multilevel histograms, histogram intersection kernel SVM (Grauman and Darrell 2005; Lazebnik et al. 2006; Maji et al. 2008) is a good and efficient choice. For very high-dimensional features, as with IFV and VLAD, linear SVM may represent a better choice (Jegou et al. 2012; Perronnin et al. 2010).

4 CNN Based Texture Representation

A large number of CNN-based texture representation methods have been proposed in recent years since the record-breaking image classification result (Krizhevsky et al. 2012)

Table 2 CNN based texture representation

Approach	Highlights
Using Pretrained Generic CNN Models (Cimpoi et al. 2016) (Sect. 4.1)	Traditional feature encoding and pooling; New pooling such as bilinear pooling (Lin and Maji 2016; Lin et al. 2018) and LFV (Song et al. 2017)
AlexNet (Krizhevsky et al. 2012)	Achieved breakthrough image classification result on ImageNet; The historical turning point of feature representation from handcrafted to CNN
VGGM (Chatfield et al. 2014; Cimpoi et al. 2016)	Similar complexity as AlexNet, but better texture classification performance
VGGVD (Simonyan and Zisserman 2015)	Much deeper than AlexNet; Much Larger model size than AlexNet and VGGM; Much better texture recognition performance than AlexNet and VGGM
GoogleNet (Szegedy et al. 2015)	Much deeper than AlexNet; Small pretrained model size; Not often used in texture classification
ResNet (He et al. 2016)	Significantly deeper than VGGVD; Smaller model size (ResNet 101) than AlexNet
Using Finetuned CNN Models (Sect. 4.2)	End-to-end learning
TCNN (Andrearczyk and Whelan 2016)	Using global average pooling; Combining outputs from multiple CONV layers
BCNN (Lin et al. 2015; Lin and Maji 2016)	Introducing a novel and orderless bilinear feature pooling method; Generalizing Fisher Vector and VLAD; Good representation ability; Very high feature dimensionality
Compact BCNN (Gao et al. 2016)	Adopting Random Maclaurin Projection or Tensor Sketch Projection to reduce the dimensionality of bilinear features (e.g. from 262144 (512^2) to 8192); Maintain similar performance to BCNN;
FASON (Dai et al. 2017)	Combining the ideas of TCNN (Andrearczyk and Whelan 2016) and Compact BCNN (Gao et al. 2016)
NetVLAD (Arandjelovic et al. 2016)	Plugging a VLAD like layer in a CNN network at the last CONV layer
DeepTEN (Zhang et al. 2017)	Similar to NetVLAD (Arandjelovic et al. 2016), integrating an encoding layer on top of CONV layers; Generalizing orderless pooling methods such as VLAD and FV in a CNN trained end to end
Texture Specific Deep Convolutional Models (Sect. 4.3)	
ScatNet (Bruna and Mallat 2013)	Use Gabor wavelets for comvolution; Mathematical interpretation of CNNs; Features being stable to deformations and preserving high frequency information;
PCANet (Chan et al. 2015)	Inspired by ScatNet (Bruna and Mallat 2013), using PCA filters to replace Gabor wavelets;Using LBP and histogramming as feature pooling; No local invariance

achieved in 2012. A key to the success of CNNs is their ability to leverage large labeled datasets to learn high quality features. Learning CNNs, however, amounts to estimating millions of parameters and requires a very large number of annotated images, an issue which rather constrains the applicability of CNNs in problems with limited training data. A key discovery, in this regard, was that CNN features pretrained on very large datasets were found to transfer well to many other problems, including texture analysis, with a relatively modest adaptation effort (Chatfield et al. 2014; Cimpoi et al. 2016; Girshick et al. 2014; Oquab et al. 2014; Sharif Razavian et al. 2014). In general, the current literature on texture classification includes examples of both employing pretrained generic CNN models or performing finetuning for specific texture classification tasks.

In this survey we will classify CNN based texture representation methods into three categories, and which form the basis of the following three sections:

- using pretrained generic CNN models,
- using finetuned CNN models, and
- using handcrafted deep convolutional networks.

These representations have had a widespread influence in image understanding; representative examples of each of these are given in Table 2.

4.1 Using Pretrained Generic CNN Models

Given the behavior of CNN transfer, the success of pretrained CNN models lies in the feature extraction and encoding steps. Similar to Sect. 3, we will describe first some commonly used networks for pretraining and then the feature extraction process.

(1) Popular Generic CNN Models can serve as good choices for extracting features, including AlexNet (Krizhevsky et al. 2012), VGGNet (Simonyan and Zisserman 2015), GoogleNet (Szegedy et al. 2015), ResNet (He et al. 2016) and DenseNet (Huang et al. 2017). Among these networks, AlexNet was proposed the earliest, and in general

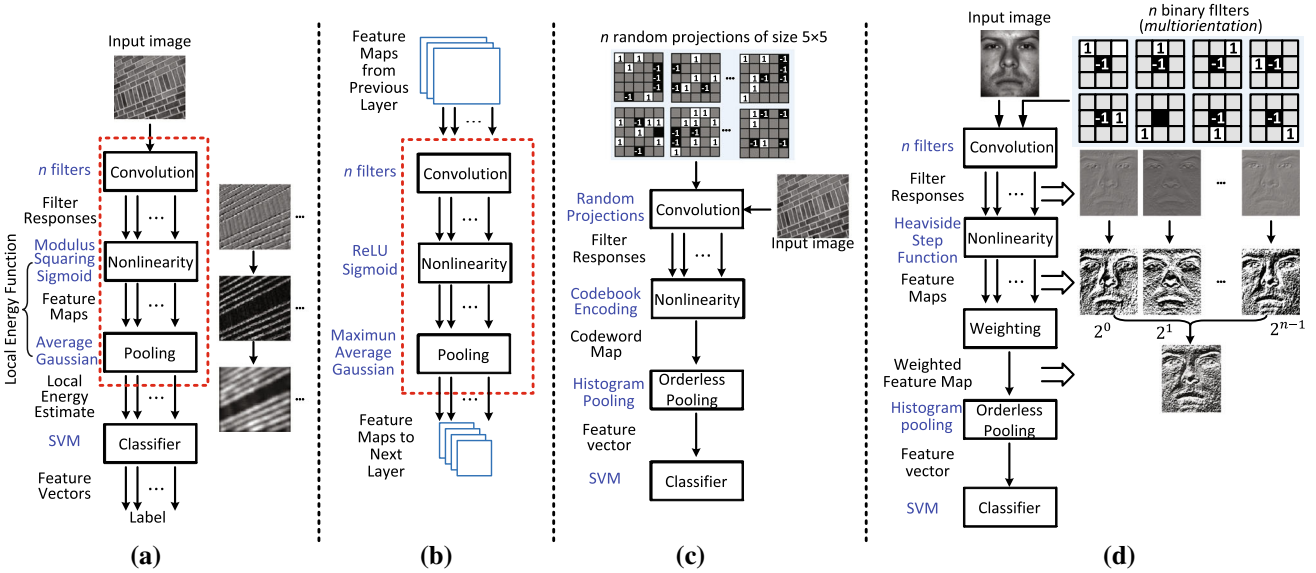


Fig. 18 Contrasting classical filtering based texture features, CNN, BoW and LBP. **a** Traditional multiscale and multiorientation filtering, **b** Basic module in Standard DCNN, **c** random projections and BoW based texture representation, **d** reformulation of the LBP using convolutional filters

the others are deeper and more complex. A full review of these networks is beyond the scope of this paper, and we refer readers to the original papers (He et al. 2016; Huang et al. 2017; Krizhevsky et al. 2012; Simonyan and Zisserman 2015; Szegedy et al. 2015) and to excellent surveys (Bengio et al. 2013; Chatfield et al. 2014; Gu et al. 2018; LeCun et al. 2015; Liu et al. 2018) for additional details. Briefly, as shown in Fig. 18b, a typical CNN repeatedly applies the following three operations:

1. Convolution with a number of linear filters,
2. Nonlinearities, such as sigmoid or rectification,
3. Local pooling or subsampling.

These three operations are highly related to traditional filter bank methods widely used in texture analysis (Randen and Husoy 1999), as shown in Fig. 18a, with the key differences that the CNN filters are learned directly from data rather than handcrafted, and that CNNs have a hierarchical architecture learning increasingly abstract levels of representation. These three operations are also closely related to the RP approach (Fig. 18c) and the LBP (Fig. 18d).

Several large-scale image datasets are usually used for CNN pretraining. Among them the commonly used ImageNet dataset, with 1000 classes and 1.2 million images (Russakovsky et al. 2015), and the scene-centric MITPlaces dataset (Zhou et al. 2014, 2018).

Comprehensive evaluations of the feature transfer effect of CNNs for the purpose of texture classification have been conducted in Cimpoi et al. (2014, 2015, 2016) and Napoletano (2017), with the following critical insights. During model transfer, features extracted from different layers exhibit different classification performance. Experiments confirm that the fully-connected layers of the CNN, whose role is primarily that of classification, tend to exhibit relatively worse generalization ability and transferability, and therefore would need retraining or finetuning on the transfer target. In contrast the convolutional layers, which act more as feature extractors, with coarser convolutional layers acting as progressively more abstract features, generally transfer well. That is, the convolutional descriptors are substantially less committed to a specific dataset than the fully connected descriptors. As a result, the source training set is relevant to classification accuracy on different datasets, and the similarity of the source and target plays a critical role when using a pretrained CNN model (Bell et al. 2015). Finally, from Cimpoi et al. (2015, 2016) and Napoletano (2017) it was found that deeper models transfer better, and that the deepest convolutional descriptors give the best performance, superior to the fully-connected descriptors, when proper encoding techniques are employed (such as FVCNN←CNN features with Fisher Vector encoder).

(2) Feature Extraction A CNN can be viewed as a composition $f_L \circ \cdots \circ f_2 \circ f_1$ of L *layers*, where the output of each layer $\mathbf{X}^l = (f_l \circ \cdots \circ f_2 \circ f_1)(\mathbf{I})$ consists of D^l feature maps of size $W^l \times H^l$. The D^l responses at each spatial location form a D^l dimensional feature vector. The network is called convolutional if all the layers are implemented as filters, in the sense that they act locally and uniformly on their input. From bottom to top layers, the image undergoes convolution, and the receptive field of these convolutional filters and the number of feature channels increases, whereas the size of

the feature maps decreases. Usually, the last several layers of a typical CNN are *fully connected* (FC) because, if seen as filters, their support is the same as the size of the input $\mathbf{X}^{l-1}$, and therefore lack locality.

The most straightforward approach to CNN based texture classification is to extract the descriptor from the fully connected layers of the network (Cimpoi et al. 2015, 2016), e.g., the FC6 or FC7 descriptors in AlexNet (Krizhevsky et al. 2012). The fully connected layers are pretrained discriminatively, which can be either an advantage or a disadvantage, depending on whether the information that they captured can be transferred to the domain of interest (Chatfield et al. 2014; Cimpoi et al. 2016; Girshick et al. 2014). The fully connected descriptors have a global receptive field and are usually viewed as global features suitable for classification with an SVM classifier. In contrast, the convolutional layers of a CNN can be used as filter banks to extract local features (Cimpoi et al. 2015, 2016; Gong et al. 2014). Compared with the global fully-connected descriptors, lower level convolutional descriptors are more robust to image transformations such as translation and occlusion. In Cimpoi et al. (2015, 2016), the features are extracted as the output of a convolutional layer, directly from the linear filters (excluding ReLU and max pooling, if any), and are combined with traditional encoders for global representation. For instance, the last convolutional layer of VGGVD (very deep with 19 layers) (Simonyan and Zisserman 2015) yields a set of 512 descriptor vectors; in Cimpoi et al. (2014, 2015, 2016) four types of CNNs were considered for feature extraction.

(3) Feature Encoding and Pooling A set of features extracted from convolutional or fully connected layers resembles a set of texture features as described in Sect. 3.2, so the traditional feature encoding methods discussed in Sect. 3.4 can be directly employed.

Cimpoi et al. (2016) evaluated several encoders, i.e. standard BoW (Leung and Malik 2001), LLC (Wang et al. 2010), VLAD (Jegou et al. 2012) and IFV (Perronnin et al. 2010) (reviewed in Sect. 3.4), for CNN features, and showed that the best performance is achieved by IFV. It has been reported that VGGVD+IFV with a linear SVM classifier produced consistently near perfect classification performance on several texture datasets: KTHTIPS (99.8%), UIUC (99.9%, UMD (99.9%) and ALOT (99.5%)), as summarized in Table 4. In addition, it obtained significant improvement on very challenging datasets like KTHTIPS2b (81.8%), FMD (79.8%) and DTD (72.3%). However, it only achieved 80.0% and 82.3% on Outex_TC10 and Outex_TC12 respectively, which are significantly worse than the near perfect performance of MRELBP on these two datasets (Liu et al. 2017); a clear indicator that DCNN based features require large amount of training samples and that they lack local invariance. Song et al. (2017) proposed a neural network to transform the FVCNN descriptors into a lower dimensional representation.

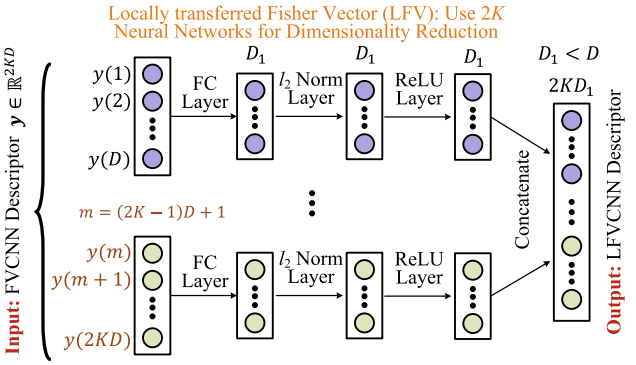


Fig. 19 Locally transferred Fisher Vector (LFV): use $2K$ neural networks for dimensionality reduction of FVCNN descriptor

As shown in Fig. 19, locally transferred FVCNN (LFVCNN) descriptors are obtained by passing the $2KD$ dimensional FVCNN descriptors of images through a multilayer neural network consisting of fully connected, l_2 normalization layers, and ReLU layers. LFVCNN achieved state of the art results on KTHTIPS2b (82.6%), FMD (82.1%) and DTD (73.8%), as shown in Table 4.

Recently, Gatys et al. (2015) showed that the Gram matrix representations extracted from various layers of VGGNet (Simonyan and Zisserman 2015) can be inverted for texture synthesis. The work of Gatys et al. (2015) ignited a renewed interest in texture synthesis (Ulyanov et al. 2017). Notably, the Gram matrix representation used in their approach is identical to the bilinear pooling of CNN features of Lin et al. (2015), which were proved to be good for texture recognition in Lin and Maji (2016). Like the traditional encoders introduced in Sect. 3.4, the bilinear feature pooling is an orderless representation of the input image and hence is suitable for modeling textures. The Bilinear CNN (BCNN) descriptors are obtained by computing the outer product of each feature $\boldsymbol{x}_i^l$ with itself, reordered into feature vectors, and subsequently pooled via sum to obtain the final global representation. The dimension of the bilinear descriptor is $(D^l)^2$, which is very high (e.g. 512^2). It was shown in Lin and Maji (2016) and Lin et al. (2018) that the texture classification performance of BCNN and FVCNN was virtually identical, indicating that bilinear pooling is as good as the Fisher vector pooling for texture recognition. It was also found that the BCNN descriptor of the last convolutional layer performed the best, in agreement with Cimpoi et al. (2016).

4.2 Using Finetuned CNN Models

Pretrained CNN models, discussed in Sect. 4.1, have achieved impressive performance in texture recognition, however training in these methods is a multistage pipeline that involves feature extraction, codebook generation, feature encoding and classifier training. Consequently, these meth-

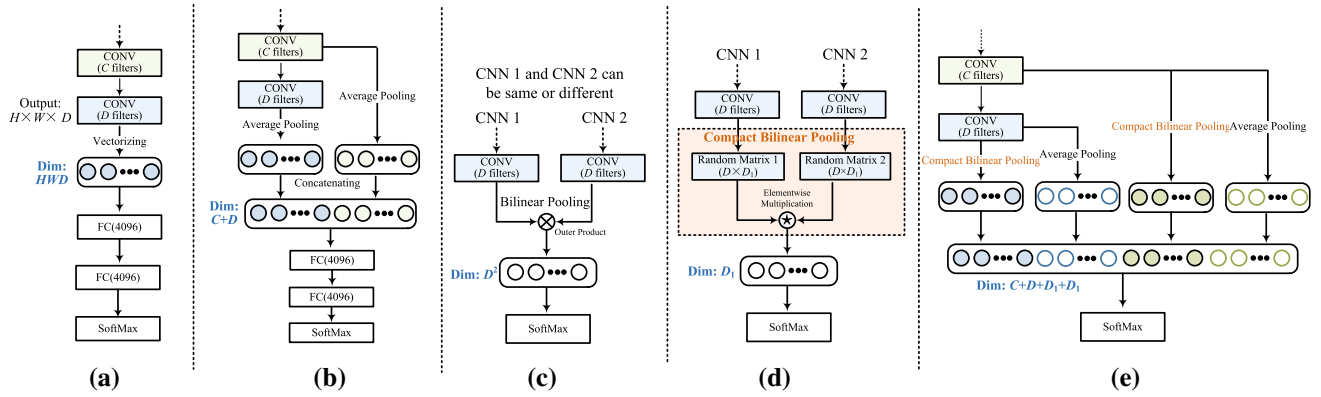


Fig. 20 Comparison of Fine Tuned CNNs: **a** standard CNN, **b** TCNN (Andrearczyk and Whelan 2016), **c** BCNN (Lin et al. 2018), **d** Compact Bilinear Pooling (Gao et al. 2016), and **e** FASON (Dai et al. 2017)

ods cannot take advantage of utilizing the full capability of neural networks in representation learning. Generally finetuning CNN models on task-specific training datasets (or learning from scratch if large-scale task-specific datasets are available) is expected to improve on already strong performance achieved by pretrained CNN models (Chatfield et al. 2014; Girshick et al. 2014). When using a finetuned CNN model, the global image representation is usually generated in an end-to-end manner; that is, the network will produce a final visual representation without additional explicit encoding or pooling steps, as illustrated in Fig. 5. When finetuning a CNN, the last fully connected layer is modified to have B nodes corresponding to the number of classes in the target dataset. The nature of the datasets used in finetuning is important to learning discriminative CNN features. The pretrained CNN model is capable of discriminating images of different objects or scene classes, but may be less effective in discerning the difference between different textures (material types) since an image in ImageNet may contain different types of textures (materials). The size of the dataset used in finetuning matters as well, since too small a dataset may be inadequate for complete learning.

To the best of our knowledge, the behaviour of a finetuned large-scale CNN like VGGNet (Simonyan and Zisserman 2015) or training it from scratch using a texture dataset have not been fully explored, almost certainly due to the fact that a large texture dataset on the scale of ImageNet (Russakovsky et al. 2015) or MITPlaces (Zhou et al. 2014) does not exist. Most existing texture datasets are small, as discussed later in Sect. 6, and according to Andrearczyk and Whelan (2016) and Lin and Maji (2016) finetuning a VGGNet (Simonyan and Zisserman 2015) or AlexNet (Krizhevsky et al. 2012) on existing texture datasets leads to negligible performance improvement. As shown in Fig. 20a, for a typical CNN like VGGNet (Simonyan and Zisserman 2015), the output of the last convolutional layer is reshaped into a single feature vector (spatially sensitive) and fed into fully connected layers (i.e., order sensitive pooling). The global spatial information is necessary for analyzing the global shapes of objects, however it has been realized (Andrearczyk and Whelan 2016; Cimpoi et al. 2016; Gatys et al. 2015; Lin and Maji 2016; Zhang et al. 2017) that it is not of great importance for analyzing textures due to the need for orderless representation. The FVCNN descriptor shows higher recognition performance than FCCNN, even if the pretrained VGGVD model is finetuned on the texture dataset (i.e., the finetuned FCCNN descriptor) (Cimpoi et al. 2016; Lin and Maji 2016). Therefore, an orderless feature pooling from the output of a convolution layer is desirable for end-to-end learning. In addition, orderless pooling does not require an input image to be of a fixed size, motivating a series of innovations in designing novel CNN architectures for texture recognition (Andrearczyk and Whelan 2016; Arandjelovic et al. 2016; Dai et al. 2017; Lin et al. 2018; Zhang et al. 2017).

A Texture CNN (TCNN) based on AlexNet, as illustrated in Fig. 20b, was developed in Andrearczyk and Whelan (2016). It simply utilizes global average pooling to transform a field of descriptor $\mathbf{X}^l \in \mathbb{R}^{W^l \times H^l \times D^l}$ at a given convolutional layer l of a CNN into a D^l dimension vector which is connected to a fully connected layer. TCNN has fewer parameters and lower complexity than AlexNet. In addition, Andrearczyk and Whelan (2016) proposed to fuse the global average pooled vector of an intermediate convolutional layer and that of the last convolutional layer via concatenation and introduced to later fully connected layers, a combination which resembles the hypercolumn feature developed in Hariharan et al. (2015). Andrearczyk and Whelan (2016) observed that finetuning a network that was pretrained on a texture-centric dataset achieves better results on other texture datasets compared to a network pretrained on an object-centric dataset of the same size, and that the size of the dataset on which the network is pretrained or finetuned predominantly influences the performance of the finetuning. These

two observations suggest that a very large texture dataset could bring a significant contribution to CNNs applied to texture analysis.

In BCNN (Lin et al. 2018), as shown in Fig. 20c, Lin et al. proposed to replace the fully connected layers with an orderless bilinear pooling layer, which was discussed in Sect. 4.1. This method was successfully applied to texture classification in Lin and Maji (2016) and obtained slightly better results than FVCNN, however the representational power of bilinear features comes at the cost of very high dimensional feature representations, which induce substantial computational burdens and require large amounts of training data, motivating several improvements on BCNN. Gao et al. (2016) proposed compact bilinear pooling, as shown in Fig. 20d, which utilizes Random Maclaurin Projection or Tensor Sketch Projection to reduce the dimensionality of bilinear representations while still maintaining similar performance to the full BCNN feature (Lin et al. 2018) with a 90% reduction in the number of learned parameters. To combine the ideas in Andrearczyk and Whelan (2016) and Gao et al. (2016), Dai et al. (2017) proposed an effective fusion network called FASON (First And Second Order information fusion Network) that combines first and second order information flow, as illustrated in Fig. 20e. These two types of features were generated from different convolutional layers and concatenated to form a single feature vector which was connected to a fully connected softmax layer for end to end training. Kong and Fowlkes (2017) proposed to represent the bilinear features as a matrix and applied a low rank bilinear classifier. The resulting classifier can be evaluated without explicitly computing the bilinear feature map which allows for a large reduction in the computational time as well as decreasing the effective number of parameters to be learned.

There are some works attempting to integrate CNN and VLAD or FV pooling in an end to end manner. In Arandjelovic et al. (2016), a NetVLAD network was proposed by plugging a VLAD-like layer into a CNN network at the last convolutional layer and allows training end to end. The model was initially designed for place recognition, however when applied to texture classification by Song et al. (2017) it was found that the classification performance was inferior to FVCNN. Similar to NetVLAD (Arandjelovic et al. 2016), a Deep Texture Encoding Network (DeepTEN) was introduced in Zhang et al. (2017) by integrating an encoding layer on top of convolutional layers, also generalizing orderless pooling methods such as VLAD and FV in a CNN trained end to end.

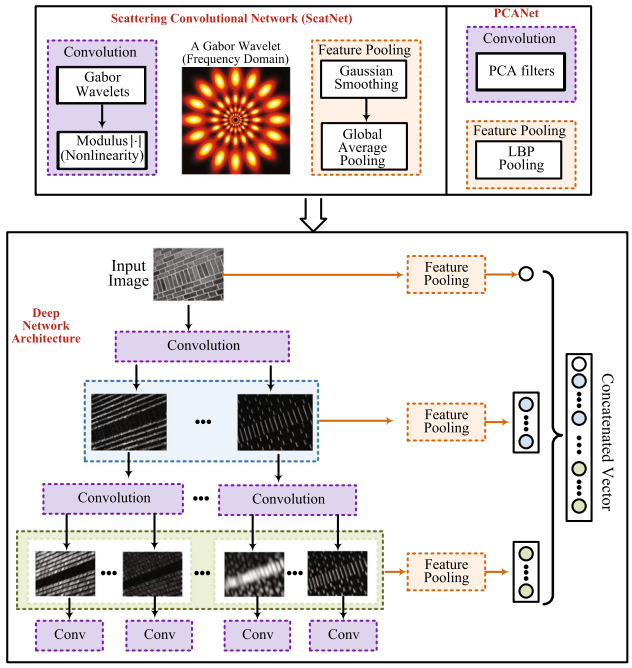


Fig. 21 Illustration of two similar handcrafted deep convolutional networks: ScatNet (Bruna and Mallat 2013) and PCANet (Chan et al. 2015)

4.3 Using Handcrafted Deep Convolutional Networks

In addition to the CNN based methods reviewed in Sects. 4.1 and 4.2, some "handcrafted"[3] deep convolutional networks (Bruna and Mallat 2013; Chan et al. 2015) deserve attention. Recall that a standard CNN architecture (as shown in Fig. 18b) consists of multiple *trainable* building blocks stacked on top of one another followed by a supervised classifier. Each block generally consists of three layers: a convolutional filter bank layer, a nonlinear layer, and a feature pooling layer. Similar to the CNN architecture, Bruna and Mallat (2013) proposed a highly influential Scattering convolution Network (ScatNet), as illustrated in Fig. 21.

The key difference from CNN, where the convolutional filters are learned from data, is that the convolutional filters in ScatNet are predetermined—they are simply wavelet filters, such as Gabor or Haar wavelets, and no learning is required. Moreover, the ScatNet usually cannot go as deep as a CNN; Bruna and Mallat (2013) suggested two convolutional layers, since the energy of the third layer scattering coefficients is negligible. Specifically, as can be seen in Fig. 21, ScatNet cascades wavelet transform convolutions with modulus nonlinearity and averaging poolers. It is shown in Bruna and Mallat (2013) that ScatNet computes

[3] Note that "handcrafted" commonly used for traditional features is somewhat imprecise, because many traditional features like Gabor filters are biologically or psychologically inspired.

translation-invariant image representations which are stable to deformations and preserve high frequency information for recognition. As shown in Fig. 21, the average pooled feature vector from each stage is concatenated to form the global feature representation of an image, which is input into a simple PCA classifier for recognition, and which has demonstrated very high performance in texture recognition (Bruna and Mallat 2013; Sifre and Mallat 2012, 2013; Sifre 2014; Liu et al. 2017). It achieved very high classification performance on Outex_TC10 (99.7%), Outex_TC12 (99.1%), KTHTIPS (99.4%), CUReT (99.8%), UIUC (99.4%) and UMD (99.7%) (Bruna and Mallat 2013; Sifre and Mallat 2013; Liu et al. 2017), but performed poorly on even challenging datasets like DTD (35.7%). A downside of ScatNet is that the feature extraction stage is very time consuming, although the dimensionality of the global representation feature is relatively low (several hundreds). ScatNet has been extended to achieve rotation and scale invariance (Sifre and Mallat 2012, 2013; Sifre 2014) and applied to other problems besides texture such as object recognition (Oyallon and Mallat 2015). Importantly, the mathematical analysis of ScatNet explains important properties of CNN architectures, and it is one of the few works that provides detailed theoretical understanding of CNNs.

Figure 21 contrasts ScatNet and PCANet, proposed by Chan et al. (2015), a very simple convolutional network based on trained PCA filters, instead of predefined Gabor wavelets, and LBP encoding (Ojala et al. 2002b) and histogramming for feature pooling. Two simple variations of PCANet, RandNet and LDANet, were also introduced in Chan et al. (2015), sharing the same topology as PCANet, but their convolutional filters are either random filters as in Liu and Fieguth (2012) or learned from Linear Discriminant Analysis (LDA). Compared with ScatNet, feature extraction in PCANet is much faster, but with weaker invariance and texture classification performance (Liu et al. 2017).

5 Attribute-Based Texture Representation

In recent years, the recognition of texture categories has been extensively studied and has shown substantial progress, partly thanks to the texture representations reviewed in Sects. 3 and 4. Despite the rapid progress, particularly with the development of deep learning techniques, we remain far from reaching the goal of comprehensive scene understanding (Krishna et al. 2017). Although the traditional goal was to recognize texture categories based on their perceptual differences or their material types, textures have other properties, as shown in Fig. 22, where we may speak of a *banded* shirt, a *striped* zebra, and a *striped* tiger. Here, *banded* and *striped* are referred to as visual texture attributes (Cimpoi et al. 2014), which describe texture patterns using human-interpretable

Fig. 22 Objects with rich textures in our daily life. Visual texture attributes like *mesh*, *spotted*, *striated*, *spotted* and *striped* provide detailed and vivid descriptions of objects

semantic words. With texture attributes, the textures shown back in Fig. 3d might all be described as *braided*, falling into a single category in the Describable Textures Dataset (DTD) database (Cimpoi et al. 2014).

The study of visual texture attributes (Bormann et al. 2016; Cimpoi et al. 2014; Matthews et al. 2013) was motivated by the significant interest raised by visual attributes (Farhadi et al. 2009; Patterson et al. 2014; Parikh and Grauman 2011; Kumar et al. 2011). Visual attributes allow the describing of objects in significantly greater detail than a category label and are therefore important towards reaching the goal of comprehensive scene understanding (Krishna et al. 2017), which would support important applications such as detailed image search, question answering, and robotic interactions. Texture attributes are an important component of visual attributes, particularly for objects that are best characterized by a pattern. It can support advanced image search applications, such as more specific queries in image search engines (e.g. a *striped* skirt, rather than just any skirt). The investigation of texture attributes and detailed semantic texture description offers a significant opportunity to close the semantic gap in texture modeling and to support applications that require fine grained texture description. Nevertheless, there are only several papers (Bormann et al. 2016; Cimpoi et al. 2014; Matthews et al. 2013) investigating the texture attributes thus far, and there is no systematic study yet attempted.

There are three essential issues in studying texture attribute based representation:

1. The identification of a universal texture attribute vocabulary that can describe a wide range of textures;
2. The establishment of a benchmark texture dataset, annotated by semantic attributes;
3. The reliable estimation of texture attributes from images, based on low level texture representations, such as the methods reviewed in Sects. 3 and 4.

Tamura et al. (1978) proposed a set of six attributes for describing textures: coarseness, contrast, directionality, line-

likeness, regularity and roughness. Amadasun and King (1989) refined this idea with the five attributes of coarseness, contrast, business, complexity, and strength. Later, Bhushan et al. (1997) studied texture attributes from the perspective of psychology, asking subjects to cluster a collection of 98 texture adjectives according to similarity and identified eleven major clusters.

Recently, inspired by the work in Bhushan et al. (1997), Farhadi et al. (2009), Parikh and Grauman (2011), Kumar et al. (2011), Matthews et al. (2013) attempted to enrich texture analysis with semantic attributes. They identified eleven commonly-used texture attributes[4] by selecting a single adjective from each of the eleven clusters identified by Bhushan et al. (1997). Then, with the eleven texture attributes, they released a publicly available human-provided labeling of over 300 classes of texture from the Outex database (Ojala et al. 2002a). For each texture image, instead of asking a subject to simply identifying the presence or absence of each texture attribute, Matthews et al. (2013) proposed a framework of pairwise comparison, in which a subject was shown two texture images simultaneously and prompted to choose the image exhibiting more of some attribute, motivated by the use of relative attributes (Parikh and Grauman 2011).

After performing a screening process on the 98 adjectives identified by Bhushan et al. (1997), Cimpoi et al. (2014) obtained a texture attribute vocabulary of 47 English adjectives and collected a dataset providing 120 example images for each attribute. They furthermore provide a comparison of BoW- and CNN-based texture representation methods for attribute estimation, demonstrating that texture attributes are excellent texture descriptors, transferring between datasets. Bormann et al. (2016) introduced a set of seventeen human comprehensible attributes (seven color and ten structural) for color texture characterization. They also collected a new database named Robotics Domain Attributes Database (RDAD) for the indoor service robotics context. They compared five low level texture representation approaches for attribute prediction, and found that not all objects can be described very well with the seventeen attributes. Clearly, which attributes are best suited for a precise description of different object and texture classes deserves further attention.

6 Texture Datasets and Performance

6.1 Texture Datasets

Datasets have played an important role throughout the history of visual recognition research. They have been one of the most important factors for the considerable progress in the field, not only as a common ground for measuring and comparing performance of competing algorithms but also pushing the field towards increasingly complicated and challenging problems. With the rapid development of visual recognition approaches, datasets have become progressively more challenging, evidenced by the fact that the recent large scale ImageNet dataset (Russakovsky et al. 2015) has enabled breakthroughs in visual recognition research. In the big data era, it becomes urgent to further enrich texture datasets to promote future research. In this section, we discuss existing texture image datasets that have been released and commonly used by the research community for texture classification, as summarized in Table 3.

The Brodatz texture database (Brodatz 1966a), derived from Brodatz (1966b), is the earliest, the most widely used and the most famous texture database. It has a relatively large number of classes (111), with each class having only one image. Many texture representation approaches exploit the Brodatz database for evaluations (Kim et al. 2002; Liu and Fieguth 2012; Ojala et al. 2002b; Pun and Lee 2003; Randen and Husoy 1999; Valkealahti and Oja 1998), however in most cases the entire database is not utilized, except in some recent studies (Georgescu et al. 2003; Lazebnik et al. 2005; Liu et al. 2017; Picard et al. 1993; Zhang et al. 2007). The database has been criticized because of the lack of intraclass variations such as scale, rotation, perspective and illumination.

The Vision Texture Database (VisTex) (Liu et al. 2005; VisTex 1995) is an early and well-known database. Built by the MIT Multimedia Lab, it has 167 classes of textures, each with only one image. The VisTex textures are imaged under natural lighting conditions, and have extra visual cues such as shadows, lighting, depth, perspective, thus closer in appearance to real-world images. VisTex is often used for texture synthesis or segmentation, but rarely for image-level texture classification.

Since 2000, texture recognition has evolved to classifying real world textures with large intraclass variations due to changes in camera pose and illumination, leading to the development of a number of benchmark texture datasets based on various real-world material instances. Among these, the most famous and widely used is the Columbia-Utrecht Reflectance and Texture (CUReT) dataset (Dana et al. 1999), with 61 different material textures taken under varying image conditions in a controlled lab environment. The effects of specularities, interreflections, shadowing, and other surface normal variations are evident, as shown in Fig. 3a. CUReT is a considerable improvement over Brodatz, where all such effects are absent. Based on the original CUReT, Varma and Zisserman (2005) built a subset for texture classification, which became the widely used benchmark to assess classification performance. CUReT has limitations of no significant scale change for most of the textures and limited in-plane

[4] Blemished, bumpy, lined, marbled, random, repetitive, speckled, spiralled, webbed, woven, and wrinkled.

Table 3 Summary of commonly-used texture databases

No.	Texture dataset	References	Total images	Texture classes	Image size	Gray or color	Imaging environment	Illumination changes	Rotation changes	Viewpoint changes	Scale changes	Image content	Instances or categories	Year	Download link
1	Brodatz	Brodatz (1966b)	111	111	640 × 640	Gray	Controlled					Objects	Instances	1966	Brodatz (1966a)
2	VisTex	–	167	167	786 × 512	Color	Wild		√			Objects	Instances	1995	VisTex (1995)
3	CUReT	Dana et al. (1999)	5612	92	200 × 200	Color	Controlled	√	Small	√		Materials	Instances	1999	CUReT (1999)
4	Outex	Ojala et al. (2002a)	8640	320	746 × 538	Color	Controlled	√	√			Materials	Instances	2002	Outex (2002)
5	KTHTIPS	Hayman et al. (2004), Fritz et al. (2004)	810	10	200 × 200	Color	Controlled	√	Small	Small	√	Materials	Instances	2004	KTHTIPS (2004)
6	UIUC	Lazebnik et al. (2005)	1000	25	640 × 480	Gray	Wild	√	√	√	√	Materials	Instances	2005	UIUC (2005)
7	KTHTIPS2a	Caputo et al. (2005), Mallikarjuna et al. (2006)	4608	11	200 × 200	Color	Controlled	√	Small	Small	√	Materials	Categories	2006	KTHTIPS (2004)
8	KTHTIPS2b	Caputo et al. (2005), Mallikarjuna et al. (2006)	4752	11	200 × 200	Color	Controlled	√	Small	Small	√	Materials	Categories	2006	KTHTIPS (2004)
9	UMD	Xu et al. (2009b)	1000	25	1280 × 960	Gray	Wild	√	√	√	√	Objects	Instances	2009	UMD (2009)
10	ALOT	Burghouts and Geusebroek (2009)	25000	250	1536 × 1024	Color	Controlled	√				Materials	Instances	2009	ALOT (2009)
11	RawFooT	Cusano et al. (2016)	3128	68	800 × 800	Color	Controlled	√				Materials	Instances	2016	Raw Food Texture (RFT) (2016)
12	FMD	Sharan et al. (2009), Sharan et al. (2013)	1000	10	512 × 384	Color	Wild	√	√			Materials	Categories	2009	FMD (2009)

Table 3 continued

No.	Texture dataset	References	Total images	Texture classes	Image size	Gray or color	Imaging environment	Illumination changes	Rotation changes	Viewpoint changes	Scale changes	Image content	Instances or categories	Year	Download link
13	DreTex	Oxholm et al. (2012)	40000	20	200 × 200	Color	Controlled	√	√		√	Materials	Instances	2012	Drexel (2012)
14	UBO2014	Weinmann et al. (2014)	1915284	7	400 × 400	Color	Synthesis	√	√	√		Materials	Categories	2014	UBO2014 (2016)
15	OpenSurfaces	Bell et al. (2013)	10422	22	Unfixed	Color	Wild	√	√	√	√	Materials	Clutter	2013	Open Surfaces (2013)
16	DTD	Cimpoi et al. (2014)	5640	47	Unfixed	Color	Wild	√	√		√	Attributes	Categories	2014	DTD (2014)
17	MINC	Bell et al. (2015)	2996674	23	Unfixed	Color	Wild	√	√	√	√	Materials	Clutter	2015	MINC (2015)
18	MINC2500	Bell et al. (2015)	57500	23	362 × 362	Color	Wild	√	√	√	√	Materials	Clutter	2015	MINC (2015)
19	GTOS	Xue et al. (2017)	34243	40	240 × 240	Color	Partially Controlled	√	√	√		Materials	Instances	2016	Ground Terrain in Outdoor Scenes (GTOS) (2016)
20	LFMD	Wang et al. (2016)	1200	12	3787 × 2632	Color	Uncontrolled	√				Materials	Categories	2016	LFMD (2016)
21	RDAD	Bormann et al. (2016)	1488	57	2592 × 1944	Color	Uncontrolled		√	√		Objects	Instances	2016	Robotics Domain Attributes Database (RDAD) (2016)

Fig. 23 Image examples from one category in KTHTIPS2

rotation. Thus, a discriminative texture feature without rotation invariance can achieve high recognition rates (Bruna and Mallat 2013).

Noticing the limited scale invariance in CUReT, researchers from the Royal Institute of Technology (KTH) introduced a dataset called "KTH Textures under varying Illumination, Pose, and Scale" (KTHTIPS) (Hayman et al. 2004; Mallikarjuna et al. 2004) by imaging ten CUReT materials at three different illuminations, three different poses, and nine different distances, but with significantly fewer settings for lighting and viewing angle than CUReT. KTHTIPS was created to extend CUReT in two directions: (i) by providing variations in scale (as shown in Fig. 23), and (ii) by imaging different samples of the CUReT materials in different settings. This supports the study of recognizing different samples of the CUReT materials; for instance, does training on CUReT enable good recognition performance on KTHTIPS? Despite pose variations, KTHTIPS rotation variations are rather limited.

Experiments with Brodatz or VisTex used different nonoverlapping subregions from the same image for training and testing; experiments with CUReT or KTHTIPS used different subsets of the images imaged from the identical sample for training and testing. KTHTIPS2 was one of the first datasets to offer considerable variations within each class. It groups textures not only by instance, but also by the type of material (e.g., wool). It is built on KTHTIPS and provides a considerable extension by imaging four physical, planar samples of each of eleven materials (Mallikarjuna et al. 2004).

The Oulu Texture (Outex) database was collected by the Machine Vision Group at the University of Oulu (Ojala et al. 2002a). It has the largest number of different texture classes (320), with each class having images photographed under three illuminations and nine rotation angles, but with limited scale variations. Based on Outex, a series of benchmark test suites were derived for evaluations of texture classification or segmentation algorithms (Ojala et al. 2002a). Among them, two benchmark datasets Outex_TC00010 and Outex_TC00012 (Ojala et al. 2002b) designated for testing rotation and illumination invariance, appear commonly in papers.

The UIUC (University of Illinois Urbana-Champaign) dataset collected by Lazebnik et al. (2005) contains 25 texture classes, with each class having 40 uncalibrated, unregistered images. It has significant variations in scale and viewpoint as well as nonrigid deformations (see Fig. 3b), but has less severe illumination variations than CUReT. The challenges of this database are that there are few sample images per class, but with significant variations within classes. Though UIUC improves over CUReT in terms of large intraclass variations, it is much smaller than CUReT both in the number of classes and the number of images per class. The UMD (University of Maryland) dataset (Xu et al. 2009b) also contains 25 texture classes; similar to UIUC, it has significant viewpoint and scale variations and uncontrolled illumination conditions. As textures are imaged under variable truncation, viewpoint, and illumination, the UIUC and the UMD have stimulated the creation of texture representations that are invariant to significant viewpoint changes.

The Amsterdam Library of Textures (ALOT) database (Burghouts and Geusebroek 2009) consists of 250 texture classes. It was collected under controlled lab environment at eight different lighting conditions. Although it has a much larger number of texture classes than UIUC or UMD, it has little scale, rotation and viewpoint variations and is therefore not a very challenging dataset. The Drexel Texture (DreTex) dataset (Oxholm et al. 2012) contains 20 different textures, each of which was imaged approximately 2000 times under different (known) illumination directions, at multiple distances, and with different in-plane and out of plane rotations. It contains stochastic and regular textures.

The Raw Food Texture database (RawFooT), has been specially designed to investigate the robustness of texture representation methods with respect to variations in the lighting conditions (Cusano et al. 2016). It consists of 68 texture classes of raw food, with each class having 46 images acquired under 46 lighting conditions which may differ in the light direction, in the illuminant color, in its intensity, or in a combination of these factors. It has no variations in rotation, viewpoint and scale.

Due to the rapid progress of texture representation approaches, the performance of many methods on the datasets described above are close to saturation, with KTHTIPS2b being an exception due to its increased complexity. However, most datasets introduced above make the simplifying assumption that textures fill images, and often there is limited intraclass variability, due to a single or limited number of instances, captured under controlled scale, viewpoint and illumination. In recent years, researchers have set their sights on more complex recognition problems where textures appear under poor viewing conditions, low resolution, and in realistic cluttered backgrounds. The Flickr Material Database (FMD) (Sharan et al. 2009, 2013) was built to address some of these limitations, by collecting many different object instances from the Internet grouped in 10 different material categories, with examples shown in Fig. 3e. The FMD (Sharan et al. 2009) focuses on identifying mate-

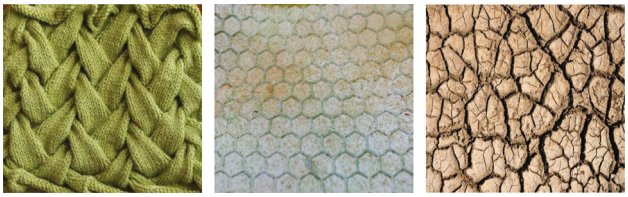


Fig. 24 Describing textures with attributes: the goal of DTD is to understand and generate automatically human interpretable descriptions such as the examples above

Fig. 25 Examples of material segments in the OpenSurfaces dataset

Fig. 26 Image samples from the MINC database. The first row are images from the *food* category, while the second row are images from *foliage*

rials such as plastic, wood, fiber and glass. The limitations of the FMD dataset is that its size is quite small, containing only 10 material classes with 100 images in each class.

The UBO2014 dataset (Weinmann et al. 2014) contains 7 material categories, with each having 12 different physical instances. Each material instance was measured by a full bidirectional texture function with 22,801 images (a sampling of 151 viewing and 151 lighting directions), resulting in a total of more than 1.9 million synthesized images. This synthesized material dataset allows classifying materials under complex real world scenarios.

Motivated by recent interests in visual attributes (Farhadi et al. 2009; Patterson et al. 2014; Parikh and Grauman 2011; Kumar et al. 2011), Cimpoi et al. (2014) identified a vocabulary of 47 texture attributes based on the seminal work of Bhushan et al. (1997) who studied the relationship between commonly used English words and the perceptual properties of textures, identifying a set of words sufficient to describing a wide variety of texture patterns. These human interpretable texture attributes can vividly characterize textures, as shown in Fig. 24. Based on the 47 texture attributes, they introduced a corresponding DTD dataset consisting of 120 texture images per attribute, by downloading images from the Internet in an effort to support directly real world applications. The large intraclass variations in the DTD are different from traditional texture datasets like CUReT, UIUC and UMD, in the sense that the images shown in Fig. 3d all belong to the *braided* class, whereas in a traditional sense these textures should belong to rather different texture categories.

Subsequent to FMD, Bell et al. (2013) released OpenSurfaces (OS) which has over 20,000 images from consumer photographs, each containing a number of high-quality texture or material segments. Images in OS have real world context, in contrast to prior databases where each image belong to one texture category and the texture fills the whole image. OS has over 100,000 segments (as shown shown in Fig. 25) that can support a variety of applications. Many, but not all, of these segments are annotated with material names, the viewpoint, reflectance, the object names and scene class. The number of segments in each material category can also be highly unbalanced in the OS.

Using the OS dataset as the seed, Bell et al. (2015) introduced a large material dataset named the Materials in Context Database (MINC) for material recognition and segmentation in the wild, with samples shown in Fig. 26. MINC has a total of 3 million material samples from 23 different material categories. MINC is more diverse, has more samples in each category, and is much larger than previous datasets. Bell et al. concluded that a large and well-sampled dataset such as MINC is key for real-world material recognition and segmentation.

Concurrent to the work by Bell et al. (2015), Cimpoi et al. (2016) derived a new dataset from OS to conduct a study of material and describable texture attribute recognition in clutter. Since not all segments in OS have a complete set of annotations, Cimpoi et al. (2016) selected a subset of segments annotated with material names, annotated the dataset with eleven texture attributes, and removed those material classes containing fewer than 400 segments. Similarly, the Robotics Domain Attributes Database (RDAD) (Bormann et al. 2016) contains 57 categories of everyday indoor object and surface textures labeled with a set of seventeen texture attributes, collected to addresses the target domain of everyday objects and surfaces that a service robot might encounter.

Wang et al. (2016) introduced a new light-field dataset of materials, called the Light-Field Material Database (LFMD). Since light-fields can capture multiple viewpoints in a single shot, they implicitly contain reflectance information, which should be helpful in material recognition. The goal of LFMD is to investigate whether 4D light-field information improves the performance of material recognition.

Finally, Xue et al. (2017) built a material database named the Ground Terrain in Outdoor Scenes (GTOS) to study the use of spatial and angular reflectance information of outdoor ground terrain for material recognition. It consists of over 30,000 images covering 40 classes o f outdoor ground terrain under varying weather and lighting conditions.

6.2 Performance

Table 4 presents a performance summary of representative methods applied to popular benchmark texture datasets. It is clear that major improvements have come from more powerful local texture descriptors such as MRELBP (Liu et al. 2017, 2016b), ScatNet (Bruna and Mallat 2013) and CNN-based descriptors (Cimpoi et al. 2016) and from advanced feature encoding methods like IFV (Perronnin et al. 2010). With the advance in CNN architectures, CNN-based texture representations have quickly demonstrated their strengths in texture classification, especially for recognizing textures with very large appearance variations, such as in KTHTIPS2b, FMD and DTD.

Off-the-shelf CNN based descriptors, in combination with IFV feature encoding, have advantages in nearly all of the benchmark datasets, except for Outex_TC10 and Outex_TC12, where texture descriptors, such as MRELBP (Liu et al. 2017, 2016b) and ScatNet (Bruna and Mallat 2013), that have rotation and gray scale invariances, give perfect accuracies, revealing one of the limitations of CNN based descriptors in being sensitive to image degradations. Despite the usual advantages of CNN based methods, it is at a cost of very high computational complexity and memory requirements. We believe that traditional texture descriptors, like the efficient LBP and robust variants such as MRELBP, still have merits in cases when real-time computation is a priority or when robustness to image degradation is needed (Liu et al. 2017).

As can be seen from Table 4, currently the highest classification scores on Outex_TC10, Outex_TC12, CUReT, KTHTIPS, UIUC, UMD and ALOT are nearly perfect, in excess of 99.5%, and quite a few texture representation approaches can achieve more than 99.0% accuracy on these datasets. Since the influential work by Cimpoi et al. (2014, 2015, 2016), who reported near perfect classification accuracies with pretrained CNN features for texture classification, subsequent representative CNN based approaches have not reported results on these datasets because performance is saturated and because the datasets are not large enough to allow finetuning to obtain improved results. The FMD, DTD and KTHTIPS2b are undoubtedly more challenging than other texture datasets, for example the UIUC and FMD texture category separation shown in Fig. 27, and these more challenging datasets appear more frequently in recent works. However, since the IFV encoding of VGGVD descriptors (Cimpoi et al. 2016), the progress on these three datasets has been slow, with incremental improvements in accuracy and efficiency obtained by building more complex or deeper CNN architectures.

As can be observed from Table 4, LBP type methods [LBP (Ojala et al. 2002b), MRELBP (Liu et al. 2016b) and BIF (Crosier and Griffin 2010)] which adopt a predefined codebook have a much more efficient feature extraction step than the remaining methods listed. For those BoW based methods which require codebook learning, since the codebook learning, feature encoding, and pooling process are similar, the distinguishing factors are the computation and feature dimensionality of the local texture descriptor. Among commonly-used local texture descriptors, those approaches first detecting local regions of interest followed by local descriptors, such as SIFT, RIFT and SPIN (Lazebnik et al. 2005; Zhang et al. 2007), are among the slowest and have relatively high dimensionality. For the CNN based methods developed in Cimpoi et al. (2014, 2015, 2016), CNN feature extraction is performed on multiple scaled versions of the original texture image, which requires more computational time. In general, CNN pretraining and finetuning is efficient, whereas CNN model training is time consuming. From Liu et al. (2017), ScatNet is computationally expensive at the feature extraction stage, though it has medium feature dimensionality. Finally, at the feature classification stage linear SVM is significantly faster than kernel SVM.

7 Discussion and Conclusion

The importance of texture representations lies in the fact that they have extended to many different problems beyond that of textures themselves. As a comprehensive survey on texture representations, this paper has highlighted the recent achievements, provided some structural categories for the methods according to their roles in feature representation, analyzed their merits and demerits, summarized existing popular texture datasets, and discussed performance for the most representative approaches. Almost any practical application is a compromise among conflicting requirements such as classification accuracy, robustness to image degradations, compactness and efficiency, number of training data available, and cost and power consumption of implementations. Although significant progress has been made, the following discussion identifies a number of promising directions for exploratory research.

Large Scale Texture Dataset Collection The constantly increasing volume of image and video data creates new opportunities and challenges. The complex variability of big image data reveals the inadequacies of conventional handcrafted texture descriptors and brings opportunities for representation learning techniques, such as deep learning,

Table 4 Performance (%) summarization of some representative methods on popular benchmark texture datasets

	Method Info			Texture Representation and Classification					Results Reported on Popular Benchmark Texture Datasets											
	Method	Published in	Described in	Local Representation	Codebook Generation	Feature Encoding	Feature Classification	Feature Dimension	Outex_TC10	Outex_TC12	Brodatz	CUReT	KTHTIPS	UIUC	UMD	ALOT	KTHTIPS2b	FMD	DTD
Traditional BoW based Texture Representation	LBP (Ojala et al. 2002b)	TAPMI 2002	Section 3.2.2	LBP^{riu2}	Predefined	BoW	Chi Square, NNC	210	99.7(◇)	92.1(◇)	90.7(◇)	97.0(◇)	–	88.4(◇)	96.2(◇)	94.2(◇)	–	–	37.1(◇)
	MR8 (Varma and Zisserman 2005)	IJCV 2005	Section 3.2.2	MR8 filters	*k*means	BoW	Chi Square, NNC	2440	–	–	–	97.4	–	–	–	–	–	–	–
	MR8 (Hayman et al. 2004)	ECCV 2004	Section 3.2.2	MR8 filters	*k*means	BoW	Chi Square, SVM	2440	–	–		98.5	–	–	–	–	–	–	–
	Lazebnik et al. (2005)	TPAMI 2005	Section 3.2.1	Corners, Blobs SPIN, RIPF	*k*means	Signatures	EMD, NNC	40 Clusters		–	88.2	72.5(⋆)	91.3(⋆)	96.0	–	–	–	–	–
	Zhang et al. (2007)	IJCV 2007	Section 3.2.1	Corners, Blobs SPIN, RIPF, SIFT	*k*means	Signatures	EMD, SVM	40 Clusters	–	–	95.4	95.3	95.5	98.7	–	–	–	–	–
	MFS (Xu et al. 2009b)	IJCV 2009	Section 3.2.2	Gaussian, Energy Laplacian	MFS Pooling		l_1, NNC	78	–	–	–	–	–	92.3	93.9	–	–	–	–
	OTF (Xu et al. 2009a)	CVPR 2009	Section 3.2.2	Multilevel Orientation Hist.	Multiscale Decomposition of MFS Vectors		RBF SVM	1160	–	–	–	–	–	97.4	98.5	–	–	–	–
	WMFS (Xu et al. 2010)	CVPR 2010	Section 3.2.2	Multiorientation Wavelet Pyramid	MFS Pooling		RBF SVM	103	–	–	–	–	–	98.6	98.7	–	–	–	–
	Patch (Varma and Zisserman 2009)	TPAMI 2009	Section 3.2.2	Patch Vectors	*k*means	BoW	Chi Square, NNC	Dependent on Dataset	–	–	92.9(⋆)	98.0	92.4(⋆)	97.8	–	–	–	–	–
	BIF (Crosier and Griffin 2010)	IJCV 2010	Section 3.2.2	BIF Features	Predefined	BoW	Chi Square, NNC	1296	–	–	–	98.6	98.5	98.8	–	–	–	–	–
	RP (Liu and Fieguth 2012)	TPAMI 2012	Section 3.2.2	Random Features	*k*means	BoW	Chi Square, NNC	Dependent on Dataset	–	–	–	98.5	–	–	–	–	–	–	–
	SRP (Liu et al. 2011a)	ICCV 2011	Section 3.2.2	SRP Features	*k*means	BoW	Chi Square, SVM	Dependent on Dataset	–	–	97.2	99.4	99.3	98.6	99.3	–	–	–	–
	Timofte and Van Gool (2012)	BMVC 2012	Section 3.2.2	BIF Features	Predefined	MultiLevel BoW	Reasoning Collaborative	1780	–	–	97.3	99.4	99.4	99.0	99.5	–	–	55.8	–
	Sharan et al. (2013)	IJCV 2013	Section 3.2.2	Eight Features (Including SIFT)	*k*means	BoW	Hist. Intersection SVM	1550	–	–	–	–	–	–	–	–	–	57.1	–
	MRELBP (Liu et al. 2016b)	TIP 2016	Section 3.2.2	$MRELBP^{riu2}$	Predefined	BoW	Chi Square, SVM	800	100(◇)	99.8(◇)	93.1(◇)	99.0(◇)	–	96.9(◇)	99.4(◇)	99.1(◇)		–	44.9(◇)
	SIFT (Lowe 2004; Cimpoi et al. 2016)	IJCV 2016	Section 3.2.2	Dense SIFT	GMM	IFV	Linear SVM	65536	–	–	–	99.0	99.5	96.6	99.1	–	70.8	59.8	58.6
CNN based Texture Representation	ScatNet (Bruna and Mallat 2013; Sifre and Mallat 2013)	TPAMI 2013 CVPR 2013	Section 4.3	Gabor Wavelet ScatNet	Gaussian Smoothing Average Pooling		PCA Classifier	596	99.7(◇)	99.1(◇)	84.5(◇)	99.8	99.4	99.4	99.7	98.3(◇)	–	–	35.7(◇)
	PCANet (Chan et al. 2015)	TIP 2015	Section 4.3	PCANet Stage 2	MultiBlock LBP Pooling		Linear SVM	32768	–	–	–	99.6	–	–	–	–	–	–	–
	AlexNet (Cimpoi et al. 2016)	IJCV 2016	Section 4.1	Pretrained CONV features (MultiScale Inputs)	GMM	IFV	Linear SVM	32768	67.3(◇)	72.3(◇)	98.2(◇)	98.5	99.6	99.2	99.7	99.1(◇)	69.7	67.2	62.9
	VGGM (Cimpoi et al. 2016)	IJCV 2016	Section 4.1		GMM	IFV	Linear SVM	65536	72.8(◇)	77.5(◇)	98.6(◇)	98.7	99.8	99.6	99.9	99.4(◇)	73.3	73.5	66.8
	VGGVD (Cimpoi et al. 2016)	IJCV 2016	Section 4.1		GMM	IFV	Linear SVM	65536	80.0(◇)	82.3(◇)	98.7(◇)	99.0	99.8	99.9	99.9	99.5(◇)	81.8	79.8	72.3
	VGGVD (Cimpoi et al. 2016)	IJCV 2016	Section 4.1	Pretrained FC features	–	–	Linear SVM	4096	–	–	–	94.5	97.9	97.0	97.7	–	75.4	77.4	62.9
	BCNN (Lin and Maji 2016)	CVPR 2016	Section 4.1	VGGVD CONV features	Bilinear Pooling		Linear SVM	262144	–	–	–	–	–	–	–	–	77.9	81.6	72.9
	LFVCNN (Song et al. 2017)	CVPR 2017	Section 4.1	VGGVD CONV features	GMM	IFV	LFV classifier	65536	–	–	–	–	–	–	–	–	82.6	82.1	73.8
	ResNet (Zhang et al. 2017)	CVPR2017	Section 4.1	ResNet50	GMM	IFV	Linear SVM	65536	–	–	–	–	–	–	–		–	78.2	–
	TCNN (Andrearczyk and Whelan 2016)	PRL 2017	Section 4.2	AlexNet CONV layers	Global Average Pooling, FC, SoftMax			4096(†)	–	–	–	99.5	–	–	–	–	73.2	–	55.8
	Compact BCNN (Gao et al. 2016)	CVPR 2016	Section 4.2	VGGVD CONV layers	Compact Bilinear Pooling (CBP), SoftMax			8192(†)	–	–	–	–	–	–	–	–	–	–	67.7
	FASON (Dai et al. 2017)	CVPR 2017	Section 4.2	VGGVD CONV layers	CBP and Global Ave. Pooling, SoftMax			9216(†)	–	–	–	–	–	–	–	–	76.4	–	72.9
	DeepTEN (Zhang et al. 2017)	CVPR 2017	Section 4.2	ResNet50	Texture Encoding Layer, SoftMax			4096(†)	–	–	–	–	–	–	–	–	–	80.2	–

All methods used the same splitting strategy for training and testing on each dataset. Specifically, for KTHTIPS2, one image per class is used for training and the remaining three for testing. For Brodatz, please see Lazebnik et al. (2005); for Outex_TC10 and Outex_TC12 please see Liu et al. (2016a). For DTD, 80 images per class are randomly selected for training and the remaining 40 for testing. For all other datasets, half of the samples per class are chosen for training and the remaining half for testing. Results are averaged over a number of random partitionings of training and testing data. All listed results are quoted from the original papers, except those marked with (⋆) from Zhang et al. (2007), and those marked (◇) from Liu et al. (2016a). For interested readers, more results on LBP variants can be found in the recent survey (Liu et al. 2016a, 2017). Those dimensions with (†) denote feature dimension before the SoftMax layer. For Brodatz, KTHTIPS2, FMD and DTD, the highest classification score is highlighted; for all other datasets classification scores higher than 99% are highlighted

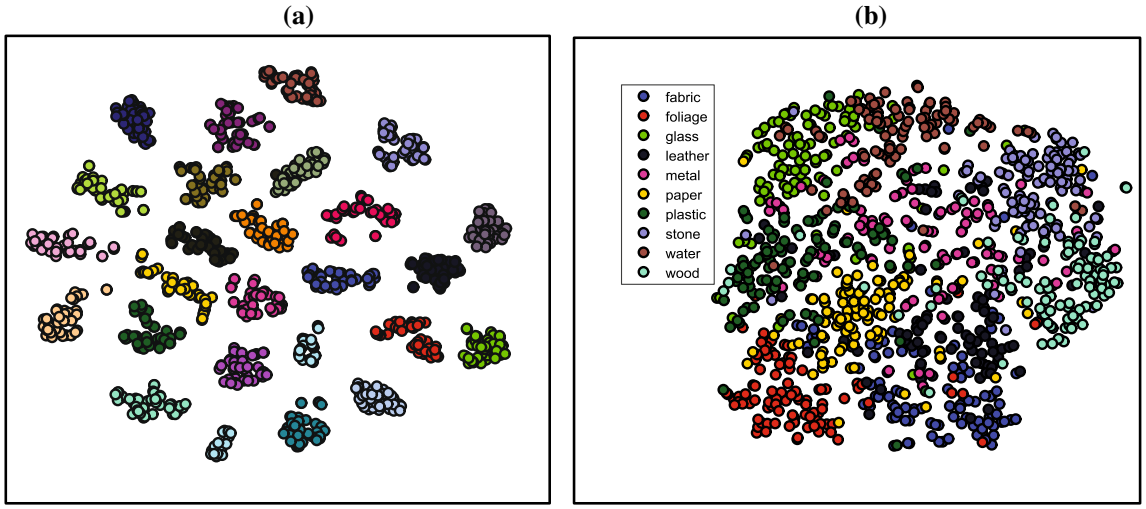


Fig. 27 t-distributed Stochastic Neighbor Embedding (tSNE) (Maaten and Hinton 2008) of textures from the IFV encoding of the VGGVD features (Cimpoi et al. 2016) from **a** the UIUC dataset (25 classes) and **b** the FMD dataset (10 classes). Clearly the classes in UIUC are more separable than those in FMD

which aim at learning good representations automatically from data. The recent success of deep learning in image classification and object recognition is inseparable from the availability of large-scale annotated image datasets such as ImageNet (Russakovsky et al. 2015) and MS COCO (Lin et al. 2014). However, deep learning based texture analysis has not kept pace with the rapid progress witnessed in other fields, partially due to the unavailability of a large-scale texture database. As a result there is significant motivation for a good, large-scale texture dataset, which will significantly advance texture analysis.

More Effective and Robust Texture Representations Despite significant progress in recent years most texture descriptors, irrespective of whether handcrafted or learned, have not been capable of performing at a level sufficient for real world textures. The ultimate goal of the community is to develop texture representations that can accurately and robustly discriminate massive image texture categories in all possible scenes, at a level comparable to the human visual system. In practical applications, factors such as significant changes in illumination, rotation, viewpoint and scale, and image degradations such as occlusions, image blur and random noise call for more discriminative and robust texture representations. Further input from psychological research of visual perception and the biology of the human visual system would be welcome.

Compact and Efficient Texture Representations There is a tension between the demands of big data and desire for highly compact and efficient feature representations. Thus, on the one hand, many existing texture representations are failing to keep pace with the emerging "big dimensionality" (Zhai et al. 2014), leading to a pressing need for new strategies in dealing with scalability, high computational complexity, and storage. On the other hand, there is a growing need for deploying highly compact and resource-efficient feature representations on platforms like low energy embedded vision sensors and handheld devices. Many of the existing descriptors would similarly fail in these contexts, and the current general trend of deep CNN architectures has been to develop deeper and more complicated networks, advances requiring massive data and power hungry GPUs, not suitable to be deployed on mobile platforms that have limited resources. As a result, there is a growing interest in building compact and efficient CNN-based features (Howard et al. 2017; Rastegari et al. 2016). While CNNs generally outperform classical texture descriptors, it remains to be seen which approaches will be most effective in resource-limited contexts, and whether some degree of LBP / CNN hybridization might be considered, such as recent lightweight CNN architectures (Lin et al. 2017; Xu et al. 2017).

Reduced Dependence on Large Amounts of Data There are many applications where texture representations are very useful and only limited amounts of annotated training data can be available, or where collecting labeled training data is too expensive (such as visual inspection, facial micro-expression recognition, age estimation and medical texture analysis). Possible research could be the development of learnable local descriptors requiring modest training data, as in Duan et al. (2018) and Lu et al. (2018), or to explore effective transfer learning.

Semantic Texture Attributes Progress in image texture representation and understanding, while substantial, has so far been mostly focused on low-level feature representation. However, in order to address advanced human-centric applications, such as detailed image search and human–robotic interaction, low-level understanding will not be sufficient.

Future efforts should be devoted to go beyond texture identification and categorization, to develop semantic and easily describable texture attributes that can be well predicted with low-level texture representations, and to explore even fine-grained and compositional structure analysis of texture patterns.

Effect of Smaller Image Size Performance evaluation of texture descriptors is usually done with texture datasets consisting of relatively large images. For a large number of applications an ability to analyze small image sizes at high speed is vital, including facial image analysis, interest region description, segmentation, defect detection, and tracking. Many existing texture descriptors would fail in this respect, and it would be important to evaluate the performance of new descriptors (Schwartz and Nishino 2015).

Acknowledgements The authors would like to thank the pioneer researchers in texture analysis and other related fields. The authors would also like to express their sincere appreciation to the associate editor and the reviewers for their comments and suggestions. This work was partially supported by the Center for Machine Vision and Signal Analysis at the University of Oulu, the Academy of Finland, Tekes Fidipro program (Grant No. 1849/31/2015), the Business Finland project (Grant No. 3116/31/2017), the Infotech Oulu, and the National Natural Science Foundation of China under Grant 61872379.

References

Aharon, M., Elad, M., & Bruckstein, A. (2006). K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, *54*(11), 4311–4322.

Ahonen, T., Hadid, A., & Pietikäinen, M. (2006a). Face decription with local binary patterns: Application to face recognition. *IEEE TPAMI*, *28*(12), 2037–2041.

Ahonen, T., Hadid, A., & Pietikainen, M. (2006b). Face description with local binary patterns: Application to face recognition. *IEEE TPAMI*, *28*(12), 2037–2041.

Ahonen, T., & Pietikäinen, M. (2007). Soft histograms for local binary patterns. In Proceedings of the finnish signal processing symposium, (Vol. 5, p. 1).

Akl, A., Yaacoub, C., Donias, M., Da Costa, J., & Germain, C. (2018). A survey of exemplar based texture synthesis methods. In *CVIU*.

Alahi, A., Ortiz, R., & Vandergheynst, P. (2012). FREAK: Fast retina keypoint. In *CVPR* (pp. 510–517).

ALOT. (2009). http://aloi.science.uva.nl/public_alot/. Accessed 16 Oct 2018.

Amadasun, M., & King, R. (1989). Textural features corresponding to textural properties. *IEEE Transactions on Systems, Man, and Cybernetics*, *19*(5), 1264–1274.

Andrearczyk, V., & Whelan, P. (2016). Using filter banks in convolutional neural networks for texture classification. *Pattern Recognition Letters*, *84*, 63–69.

Arandjelovic, R., Gronat, P., Torii, A., Pajdla, T., & Sivic, J. (2016). NetVLAD: CNN architecture for weakly supervised place recognition. In *CVPR* (pp. 5297–5307)

Baraniuk, R., Davenport, M., DeVore, R., & Wakin, M. (2008). A simple proof of the restricted isometry property for random matrices. *Constructive Approximation*, *28*(3), 253–263.

Bay, H., Tuytelaars, T., & Van Gool, L. (2006). SURF: Speeded up robust features. In *ECCV* (pp. 404–417)

Bell, S., Upchurch, P., Snavely, N., & Bala, K. (2013). Opensurfaces: A richly annotated catalog of surface appearance. *ACM Transactions on Graphics*, *32*(4), 111.

Bell, S., Upchurch, P., Snavely, N., & Bala, K. (2015). Material recognition in the wild with the materials in context database. In *CVPR* (pp. 3479–3487).

Bengio, Y., Courville, A., & Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE TPAMI*, *35*(8), 1798–1828.

Bhushan, N., Rao, A. R., & Lohse, G. L. (1997). The texture lexicon: Understanding the categorization of visual texture terms and their relationship to texture images. *Cognitive Science*, *21*(2), 219–246.

Bormann, R., Esslinger, D., Hundsdoerfer, D., Haegele, M., & Vincze, M. (2016). Texture characterization with semantic attributes: Database and algorithm. In *The 47th international symposium on robotics* (pp. 1–8).

Bosch, A., Zisserman, A., & Muñoz, X. (2008). Scene classification using a hybrid generative/discriminative approach. *IEEE TPAMI*, *30*(4), 712–727.

Boureau, Y., Ponce, J., & LeCun, Y. (2010). A theoretical analysis of feature pooling in visual recognition. In *ICML* (pp. 111–118).

Bovik, A., Clark, M., & Geisler, W. (1990). Multichannel texture analysis using localized spatial filters. *IEEE TPAMI*, *12*(1), 55–73.

Brahnam, S., Jain, L., Nanni, L., & Lumini, A. (2014). *Local binary patterns: New variants and applications*. Berlin: Springer.

Brodatz, P. (1966a). http://www.ux.uis.no/~tranden/brodatz.html. Accessed 16 Oct 2018.

Brodatz, P. (1966b). *Textures: A photographic album for artists and designers*. New York: Dover Publications.

Bruna, J., & Mallat, S. (2013). Invariant scattering convolution networks. *IEEE TPAMI*, *35*(8), 1872–1886.

Burghouts, G., & Geusebroek, J. (2009). Material specific adaptation of color invariant features. *Pattern Recognition Letters*, *30*(3), 306–313.

Calonder, M., Lepetit, V., Ozuysal, M., Trzcinski, T., Strecha, C., & Fua, P. (2012). BRIEF: Computing a local binary descriptor very fast. *IEEE TPAMI*, *34*, 1281–1298.

Candes, E. J., & Tao, T. (2006). Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE Trans Information Theory*, *52*(12), 5406–5425.

Caputo, B., Hayman, E., & Mallikarjuna, P. (2005). Class specific material categorisation. *ICCV*, *2*, 1597–1604.

Chan, T., Jia, K., Gao, S., Lu, J., Zeng, Z., & Ma, Y. (2015). PCANet: A simple deep learning baseline for image classification? *IEEE Trans Image Processing*, *24*(12), 5017–5032.

Chatfield, K., Lempitsky, V., Vedaldi, A., & Zisserman, A. (2011). The devil is in the details: an evaluation of recent feature encoding methods. In *BMVC* (Vol. 2, pp. 8).

Chatfield, K., Simonyan, K., Vedaldi, A., & Zisserman, A. (2014). Return of the devil in the details: Delving deep into convolutional nets. In *BMVC*.

Chellappa, R., & Chatterjee, S. (1985). Classification of textures using Gaussian Markov Random fields. *IEEE Trans Acoustics, Speech, and Signal Processing*, *33*(4), 959–963.

Chen, J., Shan, S., He, C., Zhao, G., Pietikainen, M., Chen, X., et al. (2010). WLD: A robust local image descriptor. *IEEE TPAMI*, *32*(9), 1705–1720.

Cimpoi, M., Maji, S., Kokkinos, I., Mohamed, S., & Vedaldi, A. (2014). Describing textures in the wild. In *CVPR* (pp. 3606–3613).

Cimpoi, M., Maji, S., Kokkinos, I., & Vedaldi, A. (2016). Deep filter banks for texture recognition, description, and segmentation. *IJCV*, *118*(1), 65–94.

Cimpoi, M., Maji, S., & Vedaldi, A. (2015). Deep filter banks for texture recognition and segmentation. In *CVPR* (pp. 3828–3836).

Cinbis, R. G., Verbeek, J., & Schmid, C. (2016). Approximate fisher kernels of non-iid image models for image categorization. *IEEE TPAMI*, *38*(6), 1084–1098.

Coates, A., & Ng, A. (2011). The importance of encoding versus training with sparse coding and vector quantization. In *ICML* (pp. 921–928).

Conners, R. W., & Harlow, C. A. (1980). A theoretical comparison of texture algorithms. *IEEE TPAMI*, *3*, 204–222.

Crosier, M., & Griffin, L. D. (2010). Using basic image features for texture classification. *IJCV*, *88*(3), 447–460.

Cross, G., & Jain, A. (1983). Markov random field texture models. *IEEE TPAMI*, *1*, 25–39.

Csurka, G., Dance, C., Fan, L., Willamowski, J., & Bray, C. (2004). Visual categorization with bags of keypoints. In *ECCV Workshop on statistical learning in computer vision*

CUReT. (1999). http://www.cs.columbia.edu/CAVE/software/curet/html/about.php. Accessed 16 Oct 2018.

Cusano, C., Napoletano, P., & Schettini, R. (2016). Evaluating color texture descriptors under large variations of controlled lighting conditions. *Journal of the Optical Society of America A*, *33*(1), 17–30.

Dahl, A., & Larsen, R. (2011). Learning dictionaries of discriminative image patches. In *BMVC*.

Dai, X., Ng, J. Y.-H., & Davis, L. S. (2017). FASON: First and second order information fusion Network for texture recognition. In *CVPR* (pp. 7352–7360).

Dana, K., Van Ginneken, B., Nayar, S., & Koenderink, J. (1999). Reflectance and texture of real world surfaces. *ACM Transactions On Graphics*, *18*(1), 1–34.

Depeursinge, A., Al-Kadi, O., & Mitchell, J. (2017). *Biomedical texture analysis*. New York: Academic Press.

Ding, C., Choi, J., Tao, D., & Davis, L. S. (2016). Multi-directional multi-level dual-cross patterns for robust face recognition. *IEEE TPAMI*, *38*(3), 518–531.

Dixit, M., Chen, S., Gao, D., Rasiwasia, N., & Vasconcelos, N. (2015). Scene classification with semantic fisher vectors. In *CVPR* (pp. 2974–2983).

Dixit, M. D., & Vasconcelos, N. (2016). Object based scene representations using fisher scores of local subspace projections. In *NIPS* (pp. 2811–2819).

Donoho, D. L. (2006). Compressed sensing. *IEEE Trans Information Theory*, *52*(4), 1289–1306.

Drexel. (2012). https://www.cs.drexel.edu/~kon/codeanddata/texture/index.html. Accessed 16 Oct 2018.

DTD. (2014). http://www.robots.ox.ac.uk/~vgg/data/dtd/. Accessed 16 Oct 2018.

Duan, Y., Lu, J., Feng, J., & Zhou, J. (2018). Context aware local binary feature learning for face recognition. *IEEE TPAMI*, *40*(5), 1139–1153.

Efros, A. A., & Leung, T. K. (1999). Texture synthesis by nonparametric sampling. *ICCV*, *2*, 1033–1038.

Everingham, M., Eslami, S., Gool, L. V., Williams, C., Winn, J., & Zisserman, A. (2015). The pascal visual object classes challenge: A retrospective. *IJCV*, *111*(1), 98–136.

Farhadi, A., Endres, I., Hoiem, D., & Forsyth, D. (2009). Describing objects by their attributes. In *CVPR* (pp. 1778–1785).

FMD. (2009). http://people.csail.mit.edu/celiu/CVPR2010/FMD/. Accessed 16 Oct 2018.

Forsyth, D., & Ponce, J. (2012). *Computer vision: A modern approach* (2nd ed.). USA: Pearson Education.

Freeman, W., & Adelson, E. (1991). The design and use of steerable filters. *IEEE TPAMI*, *13*(9), 891–906.

Fritz, M., Hayman, E., Caputo, B., & Eklundh, J. (2004). The KTH-TIPS database. http://www.nada.kth.se/cvap/databases/kth-tips/kth_tips.pdf. Accessed 16 Oct 2018.

Gao, Y., Beijbom, O., Zhang, N., & Darrell, T. (2016). Compact bilinear pooling. In *CVPR* (pp. 317–326).

Gårding, J., & Lindeberg, T. (1996). Direct computation of shape cues using scale-adapted spatial derivative operators. *IJCV*, *17*(2), 163–191.

Gatys, L., Ecker, A., & Bethge, M. (2015). Texture synthesis using convolutional neural networks. In *NIPS* (pp. 262–270).

Gatys, L., Ecker, A., & Bethge, M. (2016). Image style transfer using convolutional neural networks. In *CVPR* (pp. 2414–2423)

Georgescu, B., Shimshoni, I., & Meer, P. (2003). Mean shift based clustering in high dimensions: A texture classification example. In *ICCV* (Vol. 3, p. 456).

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR* (pp. 580–587).

Giryes, R., Sapiro, G., & Bronstein, A. M. (2016). Deep neural networks with random gaussian weights: A universal classification strategy? *IEEE Trans Signal Processing*, *64*(13), 3444–3457.

Gong, Y., Wang, L., Guo, R., & Lazebnik, S. (2014). Multi scale orderless pooling of deep convolutional activation features. In *ECCV* (pp. 392–407).

Grauman, K., & Darrell, T. (2005). The pyramid match kernel: Discriminative classification with sets of image features. *ICCV*, *2*, 1458–1465.

Griffin, L., Lillholm, M., Crosier, M., & van Sande, J. (2009). Basic image features (BIFs) arising from approximate symmetry type. In *Scale space and variational methods in computer vision* (pp. 343–355).

Griffin, L. D., & Lillholm, M. (2010). Symmetry sensitivities of derivative-of-gaussian filters. *IEEE TPAMI*, *32*(6), 1072–1083.

Ground Terrain in Outdoor Scenes (GTOS). (2016). http://computervision.engr.rutgers.edu/. Accessed 16 Oct 2018.

Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., et al. (2018). Recent advances in convolutional neural networks. *Pattern Recognition*, *77*, 354–377.

Guo, Z., Wang, X., Zhou, J., & You, J. (2016). Robust texture image representation by scale selective local binary patterns. *IEEE Trans Image Processing*, *25*(2), 687–699.

Guo, Z., Zhang, L., & Zhang, D. (2010). A completed modeling of local binary pattern operator for texture classification. *IEEE Trans Image Processing*, *9*(16), 1657–1663.

Han, J., & Ma, K. (2007). Rotation invariant and scale invariant gabor features for texture image retrieval. *Image and Vision Computing*, *25*(9), 1474–1481.

Haralick, R. (1979). Statistical and structural approaches to texture. *Proceedings of the IEEE*, *67*(5), 786–804.

Haralick, R., Shanmugam, K., & Dinstein, I. (1973). Textural features for image classification. *IEEE Trans on Systems, Man, and Cybernetics*, *6*, 610–621.

Hariharan, B., Arbeláez, P., Girshick, R., & Malik, J. (2015). Hypercolumns for object segmentation and fine-grained localization. In *CVPR* (pp. 447–456).

Hayman, E., Caputo, B., Fritz, M., & Eklundh, J. (2004). On the significance of real world conditions for material classification. In *ECCV* (pp. 253–266).

He, C., Li, S., Liao, Z., & Liao, M. (2013). Texture classification of PolSAR data based on sparse coding of wavelet polarization textons. *IEEE Trans Geoscience and Remote Sensing*, *51*(8), 4576–4590.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *CVPR* (pp. 770–778).

Howard, A., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., et al. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. In *CVPR*.

Huang, D., Shan, C., Ardabilian, M., Wang, Y., & Chen, L. (2011). Local binary patterns and its application to facial image analysis: a survey. *IEEE Transactions on Systems, Man, and Cybernetics Part C*, *41*(6), 765–781.

Huang, G., Liu, Z., Weinberger, K. Q., & van der Maaten, L. (2017). Densely connected convolutional networks. In *CVPR*.

Huang, Y., Wu, Z., Wang, L., & Tan, T. (2014). Feature coding in image classification: A comprehensive study. *IEEE TPAMI*, *36*(3), 493–506.

Jain, A., Duin, R., & Mao, J. (2000). Statistical pattern recognition: A review. *IEEE TPAMI*, *22*(1), 4–37.

Jain, A., & Farrokhnia, F. (1991). Unsupervised texture segmentation using Gabor filters. *Pattern Recognition*, *24*(12), 1167–1186.

Jegou, H., Perronnin, F., Douze, M., Sánchez, J., Perez, P., & Schmid, C. (2012). Aggregating local image descriptors into compact codes. *IEEE TPAMI*, *34*(9), 1704–1716.

Julesz, B. (1962). Visual pattern discrimination. *IRE Transactions on Information Theory*, *8*(2), 84–92.

Julesz, B. (1981). Textons, the elements of texture perception, and their interactions. *Nature*, *290*(5802), 91–97.

Julesz, B., & Bergen, J. (1983). Human factors and behavioral science: Textons, the fundamental elements in preattentive vision and perception of textures. *The Bell System Technical Journal*, *62*(6), 1619–1645.

Kadir, T., & Brady, J. (2002). Scale, saliency and scene description. Ph.D. thesis, Oxford University

Kandaswamy, U., Adjeroh, D., & Lee, M. (2005). Efficient texture analysis of SAR imagery. *IEEE Trans Geoscience and Remote Sensing*, *43*(9), 2075–2083.

Kandaswamy, U., Schuckers, S., & Adjeroh, D. (2011). Comparison of texture analysis schemes under nonideal conditions. *IEEE Trans Image Processing*, *20*(8), 2260–2275.

Keller, J., Chen, S., & Crownover, R. (1989). Texture description and segmentation through fractal geometry. *Computer Vision, Graphics, and Image Processing*, *45*(2), 150–166.

Kim, K., Jung, K., Park, S., & Kim, H. (2002). Support vector machines for texture classification. *IEEE TPAMI*, *24*(11), 1542–1550.

Kong, S., & Fowlkes, C. (2017). Low rank bilinear pooling for fine grained classification. In *CVPR* (pp. 7025–7034).

Kong, S., & Wang, D. (2012). Multilevel feature descriptor for robust texture classification via locality constrained collaborative strategy. arXiv:1203.0488

Krishna, R., Zhu, Y., Groth, O., Johnson, J., Hata, K., Kravitz, J., et al. (2017). Visual genome: Connecting language and vision using crowdsourced dense image annotations. *IJCV*, *123*(1), 32–73.

Krizhevsky, A., Sutskever, I., & Hinton, G. (2012). ImageNet classification with deep convolutional neural networks. In *NIPS* (pp. 1097–1105)

KTHTIPS. (2004). http://www.nada.kth.se/cvap/databases/kth-tips/download.html. Accessed 16 Oct 2018.

Kumar, N., Berg, A., Belhumeur, P. N., & Nayar, S. (2011). Describable visual attributes for face verification and image search. *IEEE TPAMI*, *33*(10), 1962–1977.

Kwitt, R., Vasconcelos, N., & Rasiwasia, N. (2012). Scene recognition on the semantic manifold. In *ECCV* (pp. 359–372). Springer

Lategahn, H., Gross, S., Stehle, T., & Aach, T. (2010). Texture classification by modeling joint distributions of local patterns with Gaussian mixtures. *IEEE Transaction on Image Processing*, *19*(6), 1548–1557.

Laws, K. (1980). Rapid texture identification. In Proceedings of SPIE *Conference on Image Processing for Missile Guidance* (Vol. 238, pp. 376–381).

Lazebnik, S., Schmid, C., & Ponce, J. (2003). A sparse texture representation using affine-invariant regions. In *CVPR* (vol 2), pp. II–II

Lazebnik, S., Schmid, C., & Ponce, J. (2005). A sparse texture representation using local affine regions. *IEEE TPAMI*, *27*(8), 1265–1278.

Lazebnik, S., Schmid, C., & Ponce, J. (2006). Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. *CVPR*, *2*, 2169–2178.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444.

Lee, T. S. (1996). Image representation using 2D Gabor wavelets. *IEEE TPAMI*, *18*(10), 959–971.

Leung, T., & Malik, J. (2001). Representing and recognizing the visual appearance of materials using three-dimensional textons. *IJCV*, *43*(1), 29–44.

Leutenegger, S., Chli, M., & Siegwart, R. (2011). BRISK: Binary robust invariant scalable keypoints. In *ICCV* (pp. 2548–2555)

Levi, G., & Hassner, T. (2015). Emotion recognition in the wild via convolutional neural networks and mapped binary patterns. In *ACM ICMI* (pp. 503–510)

LFMD. (2016). http://eceweb1.rutgers.edu/~kdana/code.html. Accessed 16 Oct 2018.

Li, L., Su, H., Lim, Y., & FeiFei, L. (2014). Object bank: An object level image representation for high level visual recognition. *IJCV*, *107*(1), 20–39.

Li, S. (2009). *Markov random field modeling in image analysis*. Berlin: Springer.

Lin, T., & Maji, S. (2016). Visualizing and understanding deep texture representations. In *CVPR* (pp. 2791–2799).

Lin, T., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, L. (2014). Microsoft COCO: Common objects in context. In *ECCV* (pp. 740–755).

Lin, T., RoyChowdhury, A., & Maji, S. (2015). Bilinear CNN models for fine-grained visual recognition. In *CVPR* (pp. 1449–1457).

Lin, T., RoyChowdhury, A., & Maji, S. (2018). Bilinear convolutional neural networks for fine-grained visual recognition. *IEEE TPAMI*, *40*(6), 1309–1322.

Lin, X., Zhao, C., & Pan, W. (2017). Towards accurate binary convolutional neural network. In *NIPS* (pp. 344–352).

Liu, L., & Fieguth, P. (2012). Texture classification from random features. *IEEE TPAMI*, *34*(3), 574–586.

Liu, L., Fieguth, P., Guo, Y., Wang, X., & Pietikäinen, M. (2017). Local binary features for texture classification: Taxonomy and experimental study. *Pattern Recognition*, *62*, 135–160.

Liu, L., Fieguth, P., Hu, D., Wei, Y., & Kuang, G. (2015). Fusing sorted random projections for robust texture and material classification. *IEEE TCSVT*, *25*(3), 482–496.

Liu, L., Fieguth, P., Kuang, G., & Clausi, D. (2012). Sorted random projections for robust rotation invariant texture classification. *Pattern Recognition*, *45*(6), 2405–2418.

Liu, L., Fieguth, P., Kuang, G., & Zha, H. (2011a). Sorted random projections for robust texture classification. In *ICCV* (pp. 391–398). IEEE.

Liu, L., Fieguth, P., Wang, X., Pietikäinen, M., & Hu, D. (2016a). Evaluation of LBP and deep texture descriptors with a new robustness benchmark. In *ECCV*

Liu, L., Lao, S., Fieguth, P., Guo, Y., Wang, X., & Pietikainen, M. (2016b). Median robust extended local binary pattern for texture classification. *IEEE Trans Image Processing*, *25*(3), 1368–1381.

Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X., et al. (2018). Deep learning for generic object detection: A survey. arXiv:1809.02165

Liu, L., Wang, L., & Liu, X. (2011b). In defense of soft assignment coding. In *ICCV* (pp. 2486–2493).

Liu, Y., Tsin, Y., & Lin, W. (2005). The promise and perils of near regular texture. *IJCV*, *62*(1), 145–159.

Lowe, D. (2004). Distinctive image features from scale-invariant keypoints. *IJCV*, *60*(2), 91–110.

Lu, J., Liong, V. E., & Zhou, J. (2018). Simultaneous local binary feature learning and encoding for homogeneous and heterogeneous face recognition. *IEEE TPAMI*, *40*(8), 1979–1993.

Ma, L., Tan, T., Wang, Y., & Zhang, D. (2003). Personal identification based on iris texture analysis. *IEEE TPAMI*, *25*(12), 1519–1533.

Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research*, *9*, 2579–2605.

Mairal, J., Bach, F., Ponce, J., Sapiro, G., & Zisserman, A. (2008). Discriminative learned dictionaries for local image analysis. In *CVPR* (pp. 1–8). IEEE.

Mairal, J., Ponce, J., Sapiro, G., Zisserman, A., & Bach, F. (2009). Supervised dictionary learning. In *NIPS* (pp. 1033–1040).

Maji, S., Berg, A., & Malik, J. (2008). Classification using intersection kernel support vector machines is efficient. In *CVPR* (pp. 1–8).

Malik, J., Belongie, S., Shi, J., & Leung, T. (1999). Textons, contours and regions: Cue integration in image segmentation. *ICCV*, *2*, 918–925.

Malik, J., & Perona, P. (1990). Preattentive texture discrimination with early vision mechanisms. *Journal of the Optical Society of America A: Optics, Image Science, and Vision*, *7*(5), 923–932.

Mallat, S. (1989). A theory for multiresolution signal decomposition: The wavelet representation. *IEEE TPAMI*, *11*(7), 674–693.

Mallikarjuna, P., Fritz, M., Tavakoli Targhi, A., Hayman, E., Caputo, B., et al. (2004). The KTH-TIPS and KTH-TIPS2 databases. http://www.nada.kth.se/cvap/databases/kth-tips/documentation.html. Accessed 16 Oct 2018.

Mallikarjuna, P., Tavakoli, A., Fritz, M., Hayman, E., Caputo, B., & Eklundh, J. (2006). The KTH-TIPS2 database. http://www.nada.kth.se/cvap/databases/kth-tips/kth-tips2.pdf. Accessed 16 Oct 2018.

Mandelbrot, B., & Pignoni, R. (1983). *The fractal geometry of nature*. New York: Freeman.

Manjunath, B., & Chellappa, R. (1991). Unsupervised texture segmentation using markov random field models. *IEEE TPAMI*, *13*(5), 478–482.

Manjunath, B. S., & Ma, W.-Y. (1996). Texture features for browsing and retrieval of image data. *IEEE TPAMI*, *18*(8), 837–842.

Mao, J., & Jain, A. (1992). Texture classification and segmentation using multiresolution simultaneous autoregressive models. *Pattern Recognition*, *25*(2), 173–188.

Marszałek, M., Schmid, C., Harzallah, H., J. van de W. (2007). Learning object representations for visual object class recognition. In *ICCV workshop on visual recognition challange*

Matthews, T., Nixon, M. S., & Niranjan, M. (2013) Enriching texture analysis with semantic data. In *CVPR* (pp. 1248–1255).

Mellor, M., Hong, B.-W., & Brady, M. (2008). Locally rotation, contrast, and scale invariant descriptors for texture analysis. *IEEE TPAMI*, *30*(1), 52–61.

Mikolajczyk, K., & Schmid, C. (2002). An affine invariant interest point detector. In *ECCV* (pp. 128–142).

Mikolajczyk, K., & Schmid, C. (2005). A performance evaluation of local descriptors. *IEEE TPAMI*, *27*(10), 1615–1630.

Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., et al. (2005). A comparison of affine region detectors. *IJCV*, *65*(1–2), 43–72.

MINC. (2015). http://opensurfaces.cs.cornell.edu/publications/minc/. Accessed 16 Oct 2018.

Mirmehdi, M., Xie, X., & Suri, J. (2008). *Handbook of texture analysis*. London: Imperial College Press.

Nanni, L., Lumini, A., & Brahnam, S. (2010). Local binary patterns variants as texture descriptors for medical image analysis. *Artificial Intelligence in Medicine*, *49*(2), 117–125.

Napoletano, P. (2017). Hand crafted vs learned descriptors for color texture classification. In *International workshop computational color imaging* (pp. 259–271).

Ohanian, P., & Dubes, R. (1992). Performance evaluation for four classes of textural features. *Pattern Recognition*, *25*(8), 819–833.

Ojala, T., Mäenpää, T., Pietikäinen, M., Viertola, J., Kyllönen, J., & Huovinen, S. (2002a). Outex-new framework for empirical evaluation of texture analysis algorithms. *ICPR*, *1*, 701–706.

Ojala, T., Pietikäinen, M., & Harwood, D. (1996). A comparative study of texture measures with classification based on featured distributions. *Pattern Recognition*, *29*(1), 51–59.

Ojala, T., Pietikäinen, M., & Maenpää, T. (2002b). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE TPAMI*, *24*(7), 971–987.

Ojansivu, V., & Heikkilä, J. (2008). Blur insensitive texture classification using local phase quantization. In *International conference on image and signal processing* (pp. 236–243).

Ojansivu, V., Rahtu, E., & Heikkila, J. (2008). Rotation invariant local phase quantization for blur insensitive texture analysis. In *ICPR* (pp. 1–4).

Okazawa, G., Tajima, S., & Komatsu, H. (2015). Image statistics underlying natural texture selectivity of neurons in macaque v4. *Proceedings of the National Academy of Sciences*, *112*(4), E351–E360.

Olshausen, B., & Field, D. (1996). Emergence of simple cell receptive field properties by learning a sparse code for natural images. *Nature*, *381*(6583), 607–609.

Olshausen, B. A., & Field, D. J. (1997). Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision Research*, *37*(23), 3311–3325.

Open Surfaces. (2013). http://opensurfaces.cs.cornell.edu/. Accessed 16 Oct 2018.

Oquab, M., Bottou, L., Laptev, I., & Sivic, J. (2014). Learning and transferring mid-level image representations using convolutional neural networks. In *CVPR* (pp. 1717–1724).

Outex. (2002). http://www.outex.oulu.fi/index.php?page=outex_home. Accessed 16 Oct 2018.

Oxholm, G., Bariya, P., & Nishino, K. (2012). The scale of geometric texture. In *ECCV* (pp. 58–71).

Oyallon, E., & Mallat, S. (2015). Deep roto-translation scattering for object classification. In *CVPR* (pp. 2865–2873).

Parikh, D., & Grauman, K. (2011). Relative attributes. In *ICCV* (pp. 503–510).

Patterson, G., Xu, C., Su, H., & Hays, J. (2014). The sun attribute database: Beyond categories for deeper scene understanding. *IJCV*, *108*(1–2), 59–81.

Peikari, M., Gangeh, M. J., Zubovits, J., Clarke, G., & Martel, A. L. (2016). Triaging diagnostically relevant regions from pathology whole slides of breast cancer: A texture based approach. *IEEE Transactions on Medical Imaging*, *35*(1), 307–315.

Perronnin, F., & Dance, C. (2007). Fisher kernels on visual vocabularies for image categorization. In *CVPR* (pp. 1–8).

Perronnin, F., Sanchez, J., & Mensink, T. (2010). Improving the fisher kernel for large scale image classification. *ECCV*, *6314*, 143–156.

Petrou, M., & Sevilla, P. (2006). *Image processing: Dealing with texture* (Vol. 1). Hoboken: Wiley Online Library.

Peyré, G. (2009). Sparse modeling of textures. *Journal of Mathematical Imaging and Vision*, *34*(1), 17–31.

Picard, R. W., Kabir, T., & Liu, F. (1993). Real-time recognition with the entire brodatz texture database. In *CVPR* (pp. 638–638).

Pichler, O., Teuner, A., & Hosticka, B. (1996). A comparison of texture feature extraction using adaptive Gabor filtering, pyramidal and tree structured wavelet transforms. *Pattern Recognition*, *29*(5), 733–742.

Pietikäinen, M., Hadid, A., Zhao, G., & Ahonen, T. (2011). *Computer vision using local binary patterns*. London: Springer.

Pietikäinen, M., Ojala, T., & Xu, Z. (2000). Rotation invariant texture classification using feature distributions. *Pattern Recognition*, *33*(1), 43–52.

Portilla, J., & Simoncelli, E. P. (2000). A parametric texture model based on joint statistics of complex wavelet coefficients. *IJCV*, *40*(1), 49–70.

Pun, C., & Lee, M. (2003). Log-polar wavelet energy signatures for rotation and scale invariant texture classification. *IEEE TPAMI*, *25*(5), 590–603.

Quan, Y., Xu, Y., Sun, Y., & Luo, Y. (2014). Lacunarity analysis on image patterns for texture classification. In *CVPR* (pp. 160–167).

Raad, L., Davy, A., Desolneux, A., & Morel, J. (2017). A survey of exemplar based texture synthesis. arXiv preprint arXiv:1707.07184.

Randen, T., & Husoy, J. (1999). Filtering for texture classification: A comparative study. *IEEE TPAMI*, *21*(4), 291–310.

Rasiwasia, N., & Vasconcelos, N. (2012). Holistic context models for visual recognition. *IEEE TPAMI*, *34*(5), 902–917.

Rastegari, M., Ordonez, V., Redmon, J., & Farhadi, A. (2016). XNOR-Net: ImageNet classification using binary convolutional neural networks. In *ECCV* (pp. 525–542).

Raw Food Texture (RFT). (2016). http://www.ivl.disco.unimib.it/minisites/rawfoot/download.php. Accessed 16 Oct 2018.

Reed, T., & Wechsler, H. (1990). Segmentation of textured images and gestalt organization using spatial/spatial-frequency representations. *IEEE TPAMI*, *12*(1), 1–12.

Reed, T. R., & Dubuf, J. H. (1993). A review of recent texture segmentation and feature extraction techniques. *CVGIP: Image Understanding*, *57*(3), 359–372.

Ren, J., Jiang, X., & Yuan, J. (2013). Noise resistant local binary pattern with an embedded error-correction mechanism. *IEEE Transactions on Image Processing*, *22*(10), 4049–4060.

Renninger, L. W., & Malik, J. (2004). When is scene identification just texture recognition? *Vision Research*, *44*(19), 2301–2311.

Robotics Domain Attributes Database (RDAD). (2016). http://wiki.ros.org/ipa_texture_classification. Accessed 16 Oct 2018.

Rublee, E., Rabaud, V., Konolige, K., & Bradski, G. (2011). ORB: An efficient alternative to SIFT or SURF. In *ICCV* (pp. 2564–2571).

Rubner, Y., Tomasi, C., & Guibas, L. (2000). The Earth Mover's Distance as a metric for image retrieval. *IJCV*, *40*(2), 99–121.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., et al. (2015). ImageNet large scale visual recognition challenge. *IJCV*, *115*(3), 211–252.

Ryu, J., Hong, S., & Yang, H. (2015). Sorted consecutive local binary pattern for texture classification. *IEEE Transactions on Image Processing*, *24*(7), 2254–2265.

Sanchez, J., Perronnin, F., Mensink, T., & Verbeek, J. (2013). Image classification with the fisher vector: Theory and practice. *IJCV*, *105*(3), 222–245.

Schmid, C. (2001). Constructing models for content based image retrieval. *CVPR*, *2*, 39–45.

Schwartz, G., & Nishino, K. (2015). Automatically discovering local visual material attributes. In *CVPR* (pp. 3565–3573).

Sharan, L., Liu, C., Rosenholtz, R., & Adelson, E. (2013). Recognizing materials using perceptually inspired features. *IJCV*, *103*(3), 348–371.

Sharan, L., Rosenholtz, R., & Adelson, E. (2009). Material perception: What can you see in a brief glance? *Journal of Vision*, *9*(8), 784–784.

Sharif Razavian, A., Azizpour, H., Sullivan, J., & Carlsson, S. (2014). CNN features off the shelf: An astounding baseline for recognition. In *CVPRW* (pp. 806–813).

Sharma, G., & Jurie, F. (2016). Local higher order statistics (LHS) describing images with statistics of local non-binarized pixel patterns. *CVIU*, *142*, 13–22.

Shotton, J., Winn, J., Rother, C., & Criminisi, A. (2009). Textonboost for image understanding: Multiclass object recognition and segmentation by jointly modeling texture, layout, and context. *IJCV*, *81*(1), 2–23.

Sifre, L. (2014). Rigid motion scattering for image classification, 2014. Ph.D. thesis, École Polytechnique.

Sifre, L., & Mallat, S. (2012). Combined scattering for rotation invariant texture analysis. In *Proceedings of European symposium on artificial neural networks*.

Sifre, L., & Mallat, S. (2013). Rotation, scaling and deformation invariant scattering for texture discrimination. In *CVPR* (pp. 1233–1240).

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *International conference on learning representation*.

Simonyan, K., Parkhi, O., Vedaldi, A., & Zisserman, A. (2013). Fisher vector faces in the wild. In *BMVC* (Vol. 2, p. 4).

Sivic, J., & Zisserman, A. (2003). Video google: A text retrieval approach to object matching in videos. *ICCV*, *2*, 1470–1477.

Skretting, K., & Husøy, J. (2006). Texture classification using sparse frame-based representations. *EURASIP Journal on Advances in Signal Processing*, *1*, 1–11.

Song, Y., Zhang, F., Li, Q., Huang, H., O'Donnell, L., & Cai, W. (2017). Locally transferred fisher vectors for texture classification. In *CVPR* (pp. 4912–4920).

Sulc, M., & Matas, J. (2014). Fast features invariant to rotation and scale of texture. In *ECCV* (pp. 47–62).

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. In *CVPR* (pp. 1–9).

Tamura, H., Mori, S., & Yamawaki, T. (1978). Textural features corresponding to visual perception. *IEEE Transactions on Systems, Man, and Cybernetics*, *8*(6), 460–473.

Tan, X., & Triggs, B. (2007). Enhanced local texture feature sets for face recognition under difficult lighting conditions. In *Analysis and modeling of faces and gestures* (pp. 168–182).

Timofte, R., & Van Gool, L. (2012). A training-free classification framework for textures, writers, and materials. In *BMVC* (Vol 13, p. 14).

Tuceryan, M., & Jain, A. (1993). Handbook of pattern recognition and computer vision. chap Texture Analysis (pp. 235–276).

Turner, M. (1986). Texture discrimination by gabor functions. *Biological Cybernetics*, *55*(2), 71–82.

Tuytelaars, T., & Mikolajczyk, K. (2008). Local invariant feature detectors: A survey. *Foundations and Trends in Computer Graphics and Vision*, *3*(3), 177–280.

UBO2014. (2016). http://cg.cs.uni-bonn.de/en/projects/btfdbb/download/ubo2014/. Accessed 16 Oct 2018.

UIUC. (2005). http://slazebni.cs.illinois.edu/research/uiuc_texture_dataset.zip. Accessed 16 Oct 2018.

Ulyanov, D., Vedaldi, A., & Lempitsky, V. (2017). Improved texture networks: Maximizing quality and diversity in feed forward stylization and texture synthesis. In *International conference on computer vision and pattern recognition*

UMD. (2009). http://users.umiacs.umd.edu/~fer/website-texture/texture.htm. Accessed 16 Oct 2018.

Valkealahti, K., & Oja, E. (1998). Reduced multidimensional cooccurrence histograms in texture classification. *IEEE TPAMI*, *20*(1), 90–94.

Van Gemert, J., Geusebroek, J., Veenman, C., & Smeulders, A. (2008). Kernel codebooks for scene categorization. In *ECCV* (pp. 696–709).

Van Gemert, J., Veenman, C., Smeulders, A., & Geusebroek, J.-M. (2010). Visual word ambiguity. *IEEE TPAMI*, *32*(7), 1271–1283.

Van Gool, L., Dewaele, P., & Oosterlinck, A. (1985). Texture analysis anno 1983. *Computer Vision, Graphics, and Image Processing*, *29*(3), 336–357.

Varma, M., & Garg, R. (2007). Locally invariant fractal features for statistical texture classification. In *ICCV* (pp. 1–8).
Varma, M., & Zisserman, A. (2005). A statistical approach to texture classification from single images. *IJCV*, *62*(1–2), 61–81.
Varma, M., & Zisserman, A. (2009). A statistical approach to material classification using image patches. *IEEE TPAMI*, *31*(11), 2032–2047.
Vasconcelos, N., & Lippman, A. (2000). A probabilistic architecture for content based image retrieval. *CVPR*, *1*, 216–221.
VisTex. (1995). http://vismod.media.mit.edu/vismod/imagery/VisionTexture/. Accessed 16 Oct 2018.
Wang, J., Yang, J., Yu, K., Lv, F., Huang, T., & Gong, Y. (2010). Locality-constrained linear coding for image classification. In *CVPR* (pp. 3360–3367). IEEE.
Wang, T., Zhu, J., Hiroaki, E., Chandraker, M., Efros, A. A., & Ramamoorthi, R. (2016). A 4D light field dataset and CNN architectures for material recognition. In *ECCV* (pp. 121–138).
Webb, A., & Copsey, K. (2011). *Statistical pattern recognition* (3rd ed.). New York: Wiley.
Wei, L., & Levoy, M. (2000). Fast texture synthesis using tree-structured vector quantization. In *International conference on Computer graphics and interactive techniques* (pp. 479–488).
Weinmann, M., Gall, J., & Klein, R. (2014). Material classification based on training data synthesized using a BTF database. In *ECCV* (pp. 156–171).
Weszka, J. S., Dyer, C. R., & Rosenfeld, A. (1976). A comparative study of texture measures for terrain classification. *IEEE Trans Systems, Man, and Cybernetics*, *4*, 269–285.
Winn, J., Criminisi, A., & Minka, T. (2005). Object categorization by learned universal visual dictionary. *ICCV*, *2*, 1800–1807.
Wright, J., Yang, A., Ganesh, A., Sastry, S., & Ma, Y. (2009). Robust face recognition via sparse representation. *IEEE TPAMI*, *31*(2), 210–227.
Wu, Y., Zhu, S., & Liu, X. (2000). Equivalence of julesz ensembles and FRAME models. *IJCV*, *38*(3), 247–265.
Xie, J., Hu, W., Zhu, S., & Wu, Y. (2015). Learning sparse FRAME models for natural image patterns. *IJCV*, *114*(2–3), 91–112.
Xie, X., & Mirmehdi, M. (2007). TEXEMS: Texture exemplars for defect detection on random textured surfaces. *IEEE TPAMI*, *29*(8), 1454–1464.
Xu, J., Boddeti, V. N., & Savvides, M. (2017). Local binary convolutional neural networks. In *CVPR*.
Xu, Y., Huang, S., Ji, H., & Fermuller, C. (2009a). Combining powerful local and global statistics for texture description. In *CVPR* (pp. 573–580).
Xu, Y., Ji, H., & Fermüller, C. (2009b). Viewpoint invariant texture description using fractal analysis. *IJCV*, *83*(1), 85–100.
Xu, Y., Yang, X., Ling, H., & Ji, H. (2010). A new texture descriptor using multifractal analysis in multiorientation wavelet pyramid. In *CVPR* (pp. 161–168).
Xue, J., Zhang, H., Dana, K., & Nishino, K. (2017). Differential angular imaging for material recognition. In *CVPR*.
Yang, J., Yu, K., Gong, Y., & Huang, T. (2009). Linear spatial pyramid matching using sparse coding for image classification. In *CVPR* (pp. 1794–1801).
Yang, L., Jin, R., Sukthankar, R., & Jurie, F. (2008). Unifying discriminative visual codebook generation with classifier training for object category recognition. In *CVPR* (pp. 1–8).
Ylioinas, J., Hong, X., & Pietikäinen, M. (2013). Constructing local binary pattern statistics by soft voting. In *Scandinavian conference on image analysis* (pp. 119–130).
Zhai, H., Liu, C., Dong, H., Ji, Y., Guo, Y., & Gong, S. (2015). Face verification across aging based on deep convolutional networks and local binary patterns. In *International conference on intelligent science and big data engineering* (pp. 341–350).
Zhai, Y., Ong, Y.-S., & Tsang, I. (2014). The emerging "big dimensionality". *IEEE Computational Intelligence Magazine*, *9*(3), 14–26.
Zhang, H., Jia, X., & Dana, K. (2017). Deep TEN: Texture encoding network. In *CVPR*.
Zhang, J., Marszalek, M., Lazebnik, S., & Schmid, C. (2007). Local features and kernels for classification of texture and object categories: A comprehensive study. *IJCV*, *73*(2), 213–238.
Zhang, J., & Tan, T. (2002). Brief review of invariant texture analysis methods. *Pattern Recognition*, *35*(3), 735–747.
Zhang, W., Shan, S., Gao, W., Chen, X., & Zhang, H. (2005). Local gabor binary pattern histogram sequence (LGBPHS): A novel non-statistical model for face representation and recognition. *ICCV*, *1*, 786–791.
Zhao, G., & Pietikäinen, M. (2007). Dynamic texture recognition using local binary patterns with an application to facial expressions. *IEEE TPAMI*, *29*(6), 915–928.
Zheng, L., Yang, Y., & Tian, Q. (2018). SIFT meets CNN: A decade survey of instance retrieval. *IEEE TPAMI*, *40*(5), 1224–1244.
Zhou, B., Lapedriza, A., Khosla, A., Oliva, A., & Torralba, A. (2018). Places: A 10 million image database for scene recognition. *IEEE TPAMI*, *40*(6), 1452–1464.
Zhou, B., Lapedriza, A., Xiao, J., Torralba, A., & Oliva, A. (2014). Learning deep features for scene recognition using places database. In *NIPS* (pp. 487–495).
Zhu, S. (2003). Statistical modeling and conceptualization of visual patterns. *IEEE TPAMI*, *25*(6), 691–712.
Zhu, S., Guo, C., Wang, Y., & Xu, Z. (2005). What are textons? *IJCV*, *62*(1), 121–143.
Zhu, S., Liu, X., & Wu, Y. (2000). Exploring texture ensembles by efficient markov chain monte carlo-toward a "trichromacy" theory of texture. *IEEE TPAMI*, *22*(6), 554–569.
Zhu, S., Wu, Y., & Mumford, D. (1998). Filters, random fields and maximum entropy (FRAME): Towards a unified theory for texture modeling. *IJCV*, *27*(2), 107–126.