Approches complémentaires pour une classification efficace des textures

Vu Lam Nguyen

THÈSE

présentée

à Université Paris Seine, Université Cergy-Pontoise, ENSEA, CNRS

pour obtenir le grade de :

Docteur en Science de Université Paris Seine, Université Cergy-Pontoise, ENSEA, CNRS
Spécialité : Sciences et Technologies de l'Information et de la Communication

Par

NGUYEN Vu Lam

Équipes d'accueil :
Indexation Multimédia et Intégration de Données (ETIS) – CNRS UMR 8051
École Nationale Supérieure de l'Électronique et de ses Applications

Titre de la thèse

Complementary Approaches for Efficient Texture Classification

Soutenue le 29/05/2018 devant la commission d'examen composée de :

Laurent Heutte	Professeur des Universités, Université de Rouen	Rapporteur
Philippe Carre	Professeur des Universités, Université de Poitiers	Rapporteur
Véronique Eglin	Professeur des Universités, INSA de Lyon	Examinateur
Patrick Lambert	Professeur des Universités, Université de Savoie	Examinateur
Ngoc Son Vu	MCF, ENSEA	Encadrant
Philippe-Henri Gosselin	Professeur des Universités, ENSEA	Directeur de thèse

Author's publications

[J1] **V. L. Nguyen**, N. S. Vu, H. H. Phan, P. H. Gosselin, "LBP-and-ScatNet-based combined features for efficient texture classification", *Multimedia Tools and Applications, Springer*, pp. 1-20, 2017.

[C1] **V. L. Nguyen**, N. S. Vu, P. H. Gosselin, "A Handcrafted Normalized-Convolution Network for Texture Classification", *Compact and Efficient Feature Representation and Learning in Computer Vision (CEFRL) in conjunction with ICCV, ICCV Proceedings*, October, 2017.

[C2] **V. L. Nguyen**, N. S. Vu, H. H. Phan, P. H. Gosselin, "An integrated descriptor for texture classification", *Pattern Recognition (ICPR), 23rd International Conference on. IEEE*, 2016.

[C3] **V. L. Nguyen**, N. S. Vu, P. H. Gosselin, "A scattering transform combination with local binary pattern for texture classification", *Content-Based Multimedia Indexing (CBMI), 14th International Workshop on. IEEE*, 2016. **(student best paper award)**

[J2] H. H. Phan, N. S. Vu, **V. L. Nguyen**, M. Quoy, "Action recognition based on Motion of Oriented Magnitude Patterns and feature selection", IET Computer Vision, 2018.

[C4] H. H. Phan, N. S. Vu, **V. L. Nguyen**, M. Quoy, "Motion of Oriented Magnitude Patterns for Human Action Recognition", *International Symposium on Visual Computing. Springer International Publishing*, 2016.

Résumé

Cette thèse étudie les approches complémentaires pour classer les images de texture à partir de leur apparence sans imposer de contraintes ou exiger une connaissance a priori des conditions de visualisation ou d'illumination auxquelles les images ont été prises. Les algorithmes de classification basés sur les caractéristiques sont extraits des images de texture pour les classer dans un ensemble d'étiquettes de matériaux pré-apprises.

La thèse commence par proposer une variante de modèle binaire local (LBP) pour une classification efficace des textures. Dans cette méthode proposée, une approche statistique de la représentation de la texture statique est développée. Il incorpore l'information de quantité complémentaire d'intensité d'image dans les opérateurs à base de LBP. Nous nommons notre variante LBP 'les modèles binaires entropiques locaux terminés (CLEBP) '. CLEBP capture la distribution des relations entre les mesures statistiques du caractère aléatoire des données d'image, calculées sur tous les pixels dans une structure locale. Sans aucun processus de pré-apprentissage et aucun paramètre supplémentaire à apprendre, les descripteurs CLEBP transmettent à la fois des informations globales et locales sur la texture tout en étant robustes aux variations externes. En outre, nous utilisons un filtrage biologiquement inspiré (BF) qui simule les performances de la rétine humaine en tant que technique de prétraitement. Il est montré que notre approche et la LBP conventionnelle ont la force complémentaire et qu'en combinant ces algorithmes, on obtient de meilleurs résultats que l'un ou l'autre considéré séparément.

Nous introduisons ensuite un cadre qui est une approche de combinaison de caractéristiques au problème de la classification des textures. Dans ce cadre, nous combinons les caractéristiques de LBP avec des contreparties invariantes à faible dimension, rotation et échelle, le réseau de diffusion artisanal (ScatNet). L'approche proposée est capable d'extraire des caractéristiques riches à plusieurs orientations et échelles. Les textures sont modélisées en concaténant l'histogramme des codes LBP et les valeurs moyennes des coefficients ScatNet. Nous utilisons ausi la technique de prétraitement par Filtrage Inspiré Biologiquement (BF) pour améliorer la robustesse des fonctionnalités LBP. Nous avons démontré expérimentalement que les nouvelles caractéristiques extraites du cadre proposé atteignent des performances supérieures à celles de leurs équivalents traditionnels lorsqu'elles sont comparées sur des bases de données réelles contenant de nombreuses classes présentant des variations d'imagerie significatives.

En outre, nous proposons un nouveau réseau appelé réseau de convolution normalisé. Il est inspiré par le modèle de ScatNet avec deux modifications importantes. Premièrement, la convolution normalisée remplace la convolution standard dans le modèle ScatNet pour extraire des caractéristiques de texture plus riches. Deuxièmement, au lieu d'utiliser les valeurs moyennes des coefficients du réseau, le vecteur de Fisher est exploité comme une méthode d'agrégation. Les expériences montrent

que notre réseau propose des résultats de classification compétitifs sur de nombreux benchmarks de texture difficiles.

Enfin, tout au long de la thèse, nous avons prouvé par des expériences que les approches proposées obtiennent de bons résultats de classification avec une faible ressource requise.

Abstract

This thesis investigates the complementary approaches for classifying texture images from their appearance without imposing any constraints on, or requiring any priori knowledge of, the viewing or illumination conditions at which the images were taken. Classification algorithms based on features are extracted from texture images to categorize them into a set of pre-learned material labels.

The thesis begins by proposing a Local Binary Pattern (LBP) variant for efficient texture classification. In this proposed method, a statistical approach to static texture representation is developed. It incorporates the complementary quantity information of image intensity into the LBP-based operators. We name our LBP variant 'the completed local entropy binary patterns (CLEBP)'. CLEBP captures the distribution of the relationships between statistical measures of image data randomness, calculated over all pixels within a local structure. Without any pre-learning process and any additional parameters to be learned, the CLEBP descriptors convey both global and local information about texture while being robust to external variations. Furthermore, we use biologically-inspired filtering (BF) which simulates the performance of human retina as preprocessing technique. It is shown that our approach and the conventional LBP have the complementary strength and that by combining these algorithms, one obtains better results than either of them considered separately.

We then introduce a framework which is a feature combination approach to the problem of texture classification. In this framework, we combine LBP features with low dimensional, rotation and scale invariant counterparts, the handcrafted scattering network (ScatNet). The proposed approach is capable of extracting rich features at multiple orientations and scales. Textures are modeled by concatenating histogram of LBP codes and the mean values of ScatNet coefficients. We also use the Biological Inspired Filtering (BF) preprocessing technique to enhance the robustness of LBP features. We have demonstrated by experiment that the novel features extracted from the proposed framework achieve superior performance as compared to their traditional counterparts when benchmarked on real-world databases containing many classes with significant imaging variations.

In addition, we propose a novel handcrafted network called normalized convolution network. It is inspired by the model of ScatNet with two important modification. Firstly, normalized convolution substitute for standard convolution in ScatNet model to extract richer texture features. Secondly, instead of using mean values of the network coefficients, Fisher vector is exploited as an aggregation method. Experiments show that our proposed network gains competitive classification results on many difficult texture benchmarks.

Finally, throughout the thesis, we have proved by experiments that the proposed

approaches gain good classification results with low resource required.

Contents

Chapter 1

Introduction

1.1 What is a texture?

Texture can be seen everywhere in the natural world: the surface of any visible object is textured at certain scale. Variety of textures can be observed on both artificial and natural objects such as those on wood, plants, materials and animals' skin. In a common sense, the word texture refers to the characteristic physical structure given to a material, an object, and so on, by the size, shape, arrangement, and proportions of its parts. Texture is usually described tactually in many dictionaries as smooth or rough, soft or hard, coarse or fine, matt or glossy, and so on. However, computer vision focus on its visual definition which describes the surface of objects or materials when we look at. For example, a shirt can be striped, the wings of a butterfly can be veined, and the skin of an animal can be scaly.

Therefore, textures can be broadly understood by two different points of view, tactile and visual. In the tactile aspect, textures refer to the immediate tangible feel of a surface. For instance, a surface, substance or piece of cloth feels when we touch it, how rough, smooth, hard or soft it is. While in the visual aspect, textures refer to the vision impression when we look at the objects or substance, which are related to local spatial variations of simple stimuli like color, orientation and intensity in an image. Cimpoi *et al.* [1] use a list of 47 adjectives to describe textures visually. For example, striped, veined, and scaly. This thesis focuses only on visual textures, so the term 'texture' thereafter is exclusively referred to 'visual texture' unless mentioned otherwise. Figures 1.1 and 1.2 show a few natural and man-made textures, respectively, which could be commonly seen in daily life.

Although texture is an important research area in computer vision, there is no precise formal definition related to it. The main reason is that natural textures often display different yet contradicting properties, such as regularity versus randomness, uniformity versus distortion, which can hardly be described in a consistent manner. Many researchers have been trying to define textures from a certain perspective of their nature. Haralick considers a texture as an "organized area phenomenon" which

can be decomposed into "primitives" having specific spatial distributions [2]. This definition, also known as structural approach, comes directly from human visual experience of textures. For instance, each texture in Figs 1.3 and 1.4 is the composition of particular texture elements, e.g., objects (flowers in carpets), shapes (square cells in a grid structure or six-side cells of honeycomb), veins of leaves, or simply color patterns. When these primitives are organized in a certain spatial structure, they indicate a particular placement rules to form a certain pattern. Alternatively, as Cross *et al.* formulate in [3] that a texture is "a stochastic, possibly periodic, two-dimensional image field". This definition considers texture as a stochastic process that generates it, which is also known as stochastic approach. These different definitions usually lead to alternative corresponding computational approaches to texture analysis.

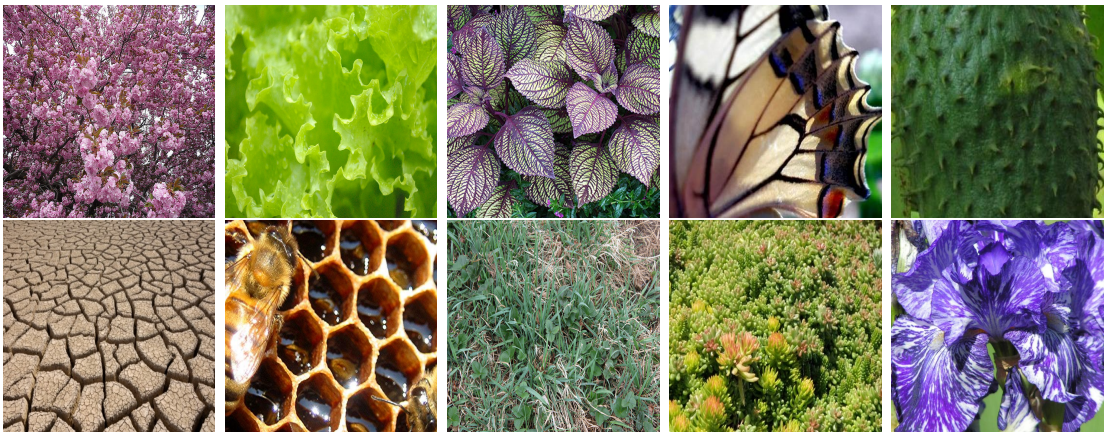


Figure 1.1: Examples of natural textures

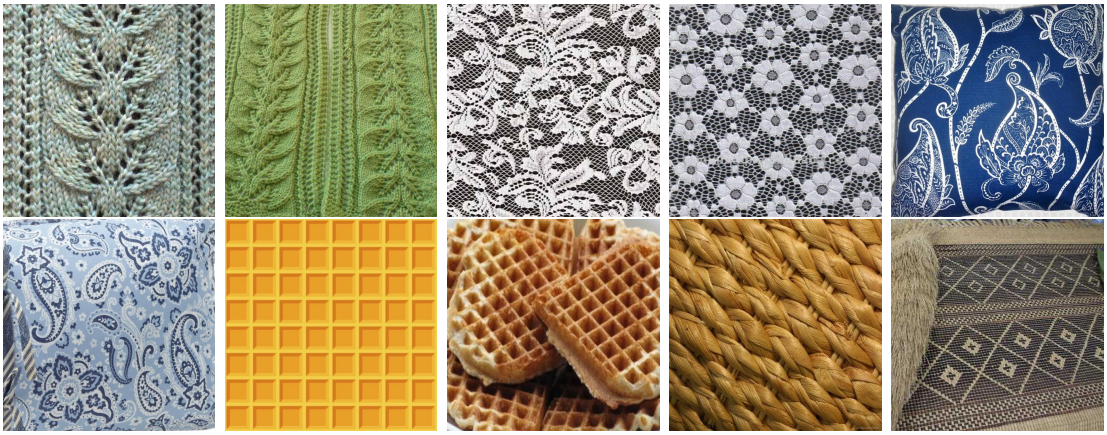


Figure 1.2: Examples of artificial textures.

However, it is an obvious consensus that spatial homogeneity is one of the most important properties a texture has. From the statistical point of view, homogeneity means statistical stationarity, i.e. that certain signal statistics of each texture region has the same values. This property relates to self-similarity: the patterns at different scales, although not identical, are represented by the same signal statistics.

A texture also exhibits local non-homogeneity, i.e. departures from strict homogeneity to some extent in a local image region. For example, in the image "flowers" in

Fig 1.1, every single flower is slightly different from another (local non-homogeneity), but as a whole they display approximate spatial uniformity and consistency (global homogeneity).

Figure 1.3: Examples of stochastic textures.

Although textures are diverse and complex, they can be separated them into categories. For example, textures can be classified into regular and stochastic ones by their degree of randomness. A regular texture is formed by fractals, easily identifiable small size elements organized into strong periodic patterns. A stochastic texture exhibits less noticeable elements and display rather random patterns. For examples, textures in Figs 1.1 and 1.3 are mostly stochastic, and those in Figs 1.4 and 1.5 are regular. Most of textures in the wild, however, are mixtures of those.

Figure 1.4: Examples of regular textures.

According to spatial homogeneity, textures can be categorized into homogeneous, weakly-homogeneous, and inhomogeneous patterns. Specifically, homogeneous texture contains ideally repetitive structures, and such uniformity produces idealized patterns (1.4). Weak homogeneity involves local spatial variation in texture elements or their spatial arrangement, which leads to more or less violates the precise repetitiveness (Fig 1.5). An “inhomogeneous texture“ mostly refers to an image without repetition and spatial self-similarity.

Figure 1.5: Examples of weakly-homogeneous textures.

1.2 Human perception of texture

Texture provides important visual clues regarding the surface properties, scenic depth, surface orientation, and etc. Human vision utilizes these information effectively in interpreting the scene and performs very efficiently texture discrimination and segmentation. It is showed in [4] that texture perception occurred in human visual system is one of the early steps of identifying objects and understanding scene.

Julesz *et al.* [5] conducted experiments as early steps toward computational texture analysis. They investigated the perceptual significance of various image statistics of texture patterns so as to determine how the human visual system responds to the variation of a specific order statistic. In the experiments, they selected synthetic textures with either repetitive or randomly placed micro-patterns such as lines, dots, and symbols. Each of those corresponds to a certain order statistic. For examples, contrast is first-order statistic, while homogeneity and curvature are second-order and third-order statistic respectively. Their works resulted in two main contributions, the conjecture and the texton theory, and subsequently become an original topic for other research.

Julesz conjectured that textures are indistinguishable for human if their second-order statistics are identical. Julesz later revised his conjecture as failure in [6], yet establishing an important idea that texture can be modeled using low-order statistics. Nowadays, texture analysis approaches represent texture with a set of sufficient statistics, which are at least conceptually based on the Julesz's conjecture.

Texton theory states that textons are "the putative units of pre-attentive human texture perception", related to texture's local features such as edges, line ends, blobs, etc. By observation, Julesz stated that the way human discriminates textures could be modeled by first-order density of such textons [5].

In addition, researches show that the performance of human retina could be simulated by a computational model using a filter bank [7]. This theory motivated mathematic models of filter-based methods which mimic human texture perception. For example, Bergen [4] proposed that textures can be decomposed into series of sub-band images using linear filter bank at different scales and orientations. Each

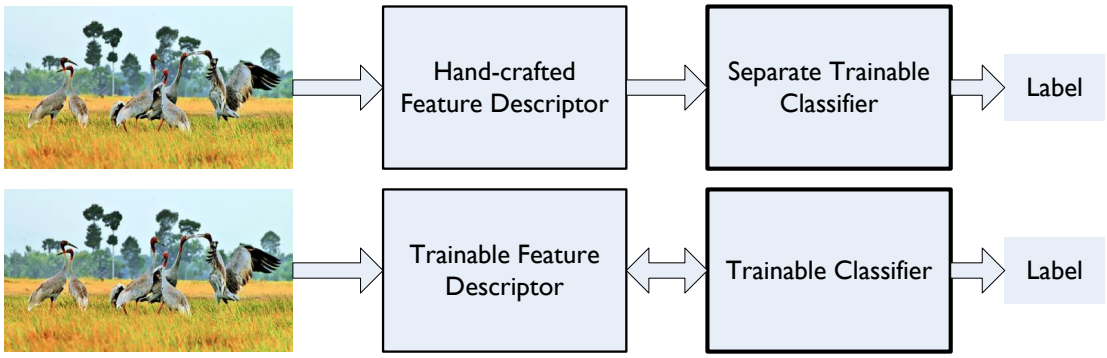


Figure 1.6: The components of a typical computer vision system (above: classical; below: recent deep learning model).

sub-band image corresponds to a particular type of features. Thus, a texture can be depicted by a distribution of the magnitude of filter responses, and therefore similarity metrics such as distances between distributions, could be utilized for discriminating textures.

To summarize, visual human perception of texture has a great impact on its analysis. Various theories have been developed to investigate the mechanisms at which human perceives textures. This leads to Julesz's conjecture. As a result, the statistical approach to texture analysis, which represents texture images by statistics of features, was proposed. The texton theory results in the structural approach which extracts texture primitives as local features. The filter-bank-based models were introduced into computational texture modeling, resulting in methods that decompose texture images by filters and transfer them into the frequency domain.

1.3 Texture analysis and its applications

Understanding, modeling and processing texture, or even simulating human visual learning process are among the major goals of texture research in computer vision. Components of a typical computer vision system can be illustrated as in Figure 1.6. Texture analysis applies different steps of the process. At the preprocessing stage, images can be simply noise disposed or more sophisticatedly segmented into contiguous regions based on texture properties of each region; At the feature extraction or feature descriptor stage (this stage can be either trainable or fixed designed), important properties of images (features) selected to provide cues for pattern classification or object identification. As a fundamental basis for all other texture-related applications, texture analysis seeks to derive a general, efficient and compact quantitative description of textures so that various mathematical operations can be used to alter, compare and transform textures. Most available texture analysis algorithms involve extracting texture features and deriving an image coding scheme for presenting selected features. These algorithms might differ in either which texture features are extracted or how they are presented in the description. For example, a statistical approach describes a texture via image signal statistics which reflect nondeterministic properties of spatial distribution of image signals. A spectral method extracts texture features from the spectral domain. A structural approach considers a texture as a hierarchy of spatial arrangements of well-defined

texture primitives. A probability model describes the underlying stochastic process that generates textures. Several representative works on texture analysis will be reviewed in more detail in Chapter 2.

Texture is a fundamental characteristics of the appearance of all natural surfaces, is ubiquitous in natural images, and is a key component of many computer vision systems. It provides important clues for identifying materials and objects, especially when shape is not available. Therefore, a wide range of applications such as industrial inspection, image retrieval, medical imaging, remote sensing, object and facial recognition can be developed depend upon analyzing textures. Four major sub domains of texture analysis are texture classification, texture segmentation, shape from texture, and texture synthesis. We start with texture classification which is the primary concern of this thesis.

1.3.1 Texture classification

The task of assigning a given texture to a pre-identified or known set of texture classes is texture classification. The majority of classification methods involve a two-stage process. The first stage is feature extraction, which yields the characteristic of each texture class in terms of feature measures. It is important to identify and select distinguishing features so that they are invariant to irrelevant transformation of the images, such as translation, rotation, and scaling. In the perfect cases, the quantitative measures of selected features should be identical or very close to each other if the texture images are identical or similar. However, it is difficult to design a universally applicable feature extractor because features are often dependent on a particular problem, and they require more or less domain knowledge to design or data to learn from. It means that there is little feature extractor which distinguishes well all categories of textures. For example, the recent state-of-the-art, FV-VGGVD [8] has a high performance on many texture datasets except the fact that it has low performance on the OUTEX suite [9, 10] because of not dealing well with the rotation of images in OUTEX. FV-VGGVD is the **F**isher **V**ector aggregation of SIFT [11] features combined with those of a pretrained very deep convolutional network developed by **V**isual **G**eometry **G**roup from University of Oxford, VGGNet [12].

At the classification stage, classifiers are trained to determine a label for each input texture based on obtained measures of selected features. In this case, a classifier is a function which takes the selected features as inputs and outputs texture class labels.

1.3.2 Texture segmentation

Partitioning an image into separate regions so that each region is homogeneous according to a particular texture characteristic is the task of texture segmentation. This consists of extracting features and deriving metrics to segregate textures. Results of segmentation can be utilized for further image processing and analysis such as for object recognition.

Depending upon whether prior knowledge regarding the texture image class is available or not, the segmentation could be regarded as supervised or unsupervised. The supervised segmentation identifies and partitions one or more regions that match texture properties shown in the training textures. Unsupervised segmentation has to first recover different texture classes from an image before separating them into regions. Compared to the supervised case, the unsupervised segmentation is more flexible for real world applications in spite of more computational expensive while it does not need the effort to set up data for training.

Texture segmentation has a variety of applications in pattern recognition and machine leaning. Texture can also be combined with other clues, such as contour information in order to segment generic images. Fig. 1.7 shows some illustrative examples. As can be seen, segmentation in itself is not always a well defined task

Figure 1.7: Segmentation of natural images using texture properties [13]. For each row, the image above is original (input), and the one below is the segmented image

since an image might have many "correct" segmentations. Nevertheless, segmentation can be a powerful tool leading on to classification and recognition. For instance, segmentation can be used as a preprocessing stage to recognize limbs and thereby detect human beings and recover their body pose. Similarly, many of the texture classification applications listed in the previous section first rely on segmentation methods to demarcate regions of interest before classifying them. As another exam-

ple, segmentation can be used to break the camouflage of hidden objects and the detected regions can then be classified. There are many other applications of being able to segment figure from ground including applications in graphics, such as image editing and background substitution , and document processing, such as extracting printed text regions

1.3.3 Texture synthesis

The task of texture synthesis is to produce a specified target texture usually from the small samples. A synthetic texture should be different from the original one, yet has perceptually identical in texture characteristics. The advantage of texture synthesis is that it can naturally handle boundary condition and avoid verbatim repetitions. This is illustrated in Fig. 1.8. Computer vision research is in favour of texture synthesis because it provides an empirical method to test texture analysis.

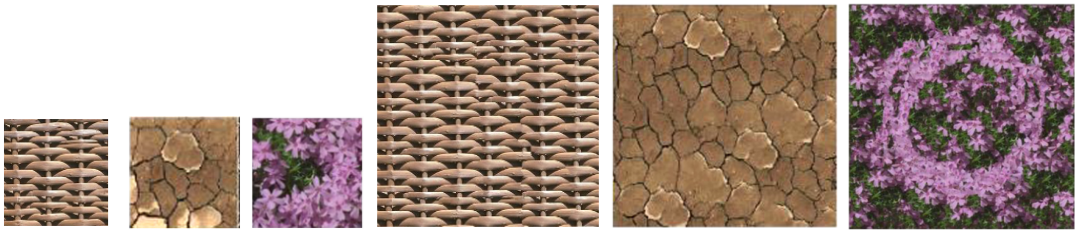


Figure 1.8: The texture synthesis problem: Given target textures on the left, the task is to synthesise similar, but not identical, output images such as the ones on the right. Results from [14].

Texture synthesis has variety of applications. In computer graphics, where there is a need to generate large amounts of realistic textures for applications such as video games and animated movies, texture synthesis methods use training images, or exemplars, to generate realistic-looking graphical textures. It can also be used together with inpainting techniques to fill in blanks left by deleting large unwanted objects in images [15]. In textile design industry, texture synthesis can be used to generate clothe patterns.

1.3.4 Shape from texture

Shape from texture is a research domain whose purpose is to estimate a 3D shape of textured objects by analyzing texture properties from their images. The shape cues can be obtained from weak homogeneity or isotropy of a texture [16]. For example, perspective projection when the surface is viewed from a slant often provides information related to texture gradient, which infers the parameters of surface shape or the underlying perspective transformation. Therefore, via properly measuring the texture gradient, a depth map and the object shapes are probably recovered.

Shape from texture has several applications. It has been used to recover the true surface orientation, to reconstruct the surface shape, and to infer the 3D layout of objects [17]. For instance, texture deformation in an image can be used as a clue to compute the plane vanish line [18], which is in turn used to affine rectify the image.

1.4 Problem statement

Texture classification is an important research area in computer vision. A great effort has been made in search of an efficient texture description.

Recent research approaches can be broadly categorized into two main branches. The first branch is learning features extracted from a convolutional neural network (CNN). They have gained wide acceptance because of their modeling power and expressiveness. Though, they also have some disadvantages. For instances, high computational cost of CNNs requires a powerful hardware like graphics processing unit (GPU) to train the model, and they need a lot of training data to build such powerful model. While second branch of texture description, the handcrafted features are not always inferior to those of the CNNs, they need less training data, and have lower computational cost compared with the learning counterparts.

However, texture discrimination is generally a difficult problem due to diversity and complexity of natural textures, there is still no single method which can extract discriminative for all type of texture data. Recent research trends follow both branches. There are works (e.g, [19]) which combine the two types of features in a system for image representation. This thesis investigates efficient handcrafted methods for texture classification.

1.5 Main contributions

Extracting robust and discriminating features are required in many applications. The most important contribution of this thesis is the complementary feature extraction approach for efficient texture classification. The main ideas of the approach is to gain both the local and global structure information of an image in order to represent it in such a way which is invariant to rotation, robust to scaling and illumination change, yet being able to preserve discriminative power.

The first contribution of this work is the development of a novel LBP variant, called the completed local entropy binary patterns (CLEBP), for texture presentation. The approach is developed by applying the LBP-based self-similarity operators on the image entropy space. This helps to incorporate the complementary information, entropy, into LBP-based features to extract both micro and macro structure information of images. Consequently, CLEBP and the conventional LBP have complementary strength and that by combining these algorithms, one obtains better results than either of them considered separately. Experimental results on four large texture databases, including Outex, KTHTIPS2b, CuRet and UIUC show that our approach is more efficient than many other LBP variants. Although CLEBP is a micro-macro complementary feature approach, the supplementary components are formed by the same type of micro-feature LBP encoding method. The next contribution introduces an alternative approach texture representation, a compound framework formed by combining two complementary types of features results in high discriminative texture method.

The second contribution is the set up of a framework from which two type of not-concurrent yet complementary descriptors, LBP-based and ScatNet, are combined. In this work, we empirically prove that these types of features when combined can capture better both local and global structure information of texture images. The

method obtains very competitive classification results on four challenging datasets, Outex, KTHTIPS2b, CuRet and UIUC. It is worth noticing that we combine CLBP (a basic LBP method) rather than CLEBP (a novel advanced method) and ScatNet because we want to prove in this contribution that even a micro LBP and macro structure feature ScatNet can be complementary. Advanced LBPs combined with ScatNet is intended for future works. With a different approach, next contribution introduces a novel ScatNet variant which enhances the strength of ScatNet for texture representation, namely a Normalized-Convolution Network.

The third contribution is the introduction of a novel convolution network, so-called Normalized-Convolution Network. This work is inspired by the model of ScatNet [20] with two important modifications. 1) Normalized Convolution operator [21] substitute for the standard convolution. Normalized Convolution is based on the the signal/certainty philosophy, i.e. separating the values of a signal from the certainty of the measurements. 2) The implementation [22] of Fisher Vector aggregation of the network coefficients is applied instead of the mean values of those. The proposed method not only gains high competitive results on the challenging texture databases: UIUC, KTH-TIPS2 but also poses the potentiality to high performance on other challenging texture data such as occlusion or noised data.

1.6 Thesis outline

The remainder of the thesis is organized as follows. Chapter 2 presents a survey of the literature and places the current versions of the classification and synthesis problems in a historical perspective. It is begun with discussions on a brief history over the last decades or so and recalls how the classification problem has evolved from binary pattern discrimination to 2D texture classification and finally to 3D texture classification. Solutions of using filter banks and co-occurrence have coped by building more and more complex representations and this evolution is charted as well. Filter bank based methods are mainly emphasized because they have provided key breakthroughs and insights into the field. Since filters have played such an important role, work on designing optimal filter banks for texture classification is reviewed in the chapter. In addition, details of four popular texture databases, CUReT, KTH-TIPS, Outex, and UIUC, are presented. These datasets are used for all experiments in this thesis.

Chapter 3 introduces Local Binary Pattern (LBP), the relation of LBP to earlier texture feature methods, as well as its variants. The chapter begins by formulating the calculation method of the original LBP operator [23]. Then, the measures for dealing with image rotation, dimensionality reduction (uniform LBP), and image scaling are presented. Next, the relationship between LBP and the earlier texture methods is discussed. At the end of the chapter, we summarize and classify LBP variants into groups for easily understanding this simple yet efficient method.

Chapter 4 reviews the scattering transform, its construction, and the scattering network (ScatNet) implementation. Scattering transform is actually a cascade of wavelet transforms followed by a modulus non-linearity. Therefore, wavelet transform methods such as Gabor, Morlet and their implementation are also reassessed in this chapter. Convolutional neural network (CNN) method is also presented in

the chapter with some of its examples because CNN and ScatNet to a certain extend are related.

Chapter 5 proposes a novel LBP variant for texture representation. It is built by applying the LBP-based self-similarity operators upon the image entropy space. Our LBP-based method encodes the relationships between the statistical measure of the complementary information, the randomness of intensities, i.e., entropy, within a local image structure. Motivated by the completed LBP operators proposed by Guo *et al.* [24], we include both the sign and magnitude components of the difference between the quantity information of a given local image patch and its surrounding counterparts, and the local entropy of the considered patch. We figure out how our proposed method, the CLEBP descriptors, without any pre-learning process and any additional parameters to be learned, convey both local and global information about texture in this chapter. By experiments, we show that our approach and the conventional LBP have complementary strength and that by combining these algorithms, one obtains better results than either of them considered separately. Classification results on four texture datasets are reported, and compared to state of the art.

In chapter 6, we propose another type supplementary feature method, the LBP and ScatNet based complementary approach. Similar to the strategy in chapter 5, the final features of this method are the combination of micro and macro structure information. However, the approach in chapter 5 is the supplement of the similar type of features while LBP and ScatNet combination presented in this chapter come from two different families of features. We set up a framework, and validate its sufficient for classification and experimentally verify on four challenging texture datasets: UIUC, CURet, Outex, and KTH-TIPS-2b.

In chapter 7, we propose novel method, so-called Normalized-Convolution Network (NmzNet) for texture classification. Its implementation, by computing successively normalized convolution [21] with a predefined filter bank (Gabor filter bank) and modulus non-linearities, is presented. The chapter also presents Fisher Vector aggregation method which is used to aggregate coefficients from NmzNet's layers for final discriminative features. The results of experimental evaluation on three texture datasets UIUC, KTH-TIPS-2a, and KTH-TIPS-2b are reported and compared to state-of-the-arts.

We end in chapter 8 by discussing some of the conclusions that can be drawn from this thesis and exploring some of the avenues for future work.

Chapter 2

Literature Review

This chapter reviews the evolution of texture classification problem over the last decades and highlight some important achievements and innovations made in texture classification as well as texture analysis problem. Section 2.1 presents an overview of the field while section 2.2 discusses the texture datasets that have been used in this thesis to benchmark performance.

2.1 Overview

The pioneering work on the visual perception of texture [25] formed the basis for a lot of the subsequent research that followed in texture analysis. Since then, many various approaches have been formulated. There is a wide variety of techniques for texture description have been proposed. They are generally divided into four categories: statistical, geometrical, model-based and signal processing. Among the most well-known traditional approaches are statistical methods based on co-occurrence matrices of second order gray level statistics [26] or first order statistics of local property values (difference histograms) [27], signal processing methods based on local linear transforms, multichannel Gabor filtering or wavelets [28], and model-based methods based on fractals or Markov random fields [3].

From the mid 1990s to the mid 2000s, many discriminative and computationally efficient local texture descriptor methods were proposed. For example, local binary patterns (LBP) [29], which has led to a certain progress in applying texture methods to various computer vision problems. The focus of the research has broadened from 2D textures to 3D textures such as compact representation of bidirectional texture functions, three-dimensional textons [30], statistical distribution of rotationally invariant clustered filter responses [31], and spatiotemporal (dynamic) textures [32].

Since the late 2000s and afterwards have been witnessing significant progress imagery research in general and texture analysis in specific, with the several new handcrafted features and deep learning convolutional neural network methods proposed. These new innovation can be a long list, but the recent well-known state-of-the-art in the field of texture analysis such as LBP variants [33, 34], scattering

network (ScatNet) [35], and convolutional neural network (CNN) methods [8] cannot be absent.

In this section, we present an overview of some of these approaches. The aim is to put into perspective some of the important research that has shaped our understanding of texture as it is today. We begin with discussions on a brief history over the last decades or so and review how the classification problem has evolved from binary pattern discrimination to 2D texture classification and finally to 3D texture classification. Proposed solutions using filter banks and co-occurrence have coped by building more and more complex representations and this evolution is charted as well. In subsection 2.1.1, the emphasis is primarily on filter bank based methods as they have provided key breakthroughs and insights into the field. Since filters have played such an important role, work on designing optimal filter banks for texture classification is reviewed next in subsection 2.1.2. Finally, the convolutional neural network (CNN) method is presented in the field of texture analysis.

2.1.1 A brief history of texture descriptive methods

Many early works on texture classification was concerned with Julesz's conjecture [6] that two textures were not perceptually distinguishable if they had identical second order statistics (i.e. the distribution of intensities over pairs of pixels was identical). Efforts to prove or disprove the supposition focused on whether binary image patterns, such as the ones shown in Figure 2.1, were preattentively discriminable by the human eye or not. The conjecture was later disproved by Julesz himself and replaced by a statistical theory of textons [5]. It was posited that textons were basic texture primitives such as line intersections, corners, terminators etc., and two textures having different level of texton densities can be easily distinguished.

At this time, the classification task was simply to separate textures (binary images) formed by the repetitive fundamental micro patterns. Therefore, a lot of efforts were made on textures of man-made surface while little attention was paid to real world textures except the work on Brodatz album [37]. The theory of textons lasted until late 1980s when there are two other major proposals. The first one was about how to formalize a list of universal textons. The second was how to generalize the theory to gray scale images. At the same time, researchers had been experimenting with using filter banks for texture analysis throughout the decade [38]. In fact, Julesz used filter responses as a means to explain counter examples of his second order statistics theory [6] and then later postulated a connection between textons and filter banks [39]. Because the limitations of the texton theory were realized, filter bank based methods started gaining popularity. Two very influential theories which drew attention away from textons and sparked an alternate interest in filter banks were developed in [40, 41]. Bergen *et al.* showed in [40] that the responses of size tuned center surround filters can be used to discriminate textures. For example, as illustrated in Figure 2.2, if the size of the primitives increases (decreases) then the density of terminators and crossings remains unchanged whereas humans find it easier (harder) to distinguish the two textures. Such phenomenon is similar to the distribution of filter responses as the difference between the total energies of the two

Figure 2.1: Texture pairs with identical second-order statistics. The bottom halves of the images consist of texture tokens that are different from the ones in the top half. (a) Humans cannot perceive the two regions without careful examination. (b) The two different regions are immediately discriminable by humans. [36]

distributions increases or decreases according to human perception. The results of work in [42] supported this theory when they showed how the texture primitives of natural gray scale images can be computed by thresholding the responses of center surround filters followed by morphological operations. It was also shown that texture boundaries could be computed for natural images. Inspired by human visual system, Malik *et al.* [41] developed a bank of even-symmetric linear filters followed by half-wave rectification to give a set of responses modeling outputs of primary visual cortex simple cells. Their model can predict the salience of texture boundaries in any arbitrary gray-scale image. They proved by experiment on psychophysical data that their model can predict well the degree of texture discriminability as human observers can do. Later, there are also works [43] with a certain influence. They provided methods of calculating filter responses at all possible orientations and scales from a small basis set.

At the time, filter banks preferred to texton feature methods because they could be used to analyze gray scale images. This led to classification of 2D texture being available. The emphasis shifted from distinguishing patterns in binary images of man-mad textures to classifying gray scale images of real world textures. Because of computational limitation, the early filter bank based methods only used low order moments to characterize the distribution of filter responses. The feature extraction scheme was to form a long vector whose components formed by the mean or variance of individual filter response distributions. Some techniques were also used for processing filter responses such as rectification, energy measurement or conversion to a rotationally invariant frame. A classifiers were trained on the feature vectors and used to classify novel images. Typical examples of such frameworks were [44, 45]. Performance was generally assessed on variations of the Brodatz album [46] and

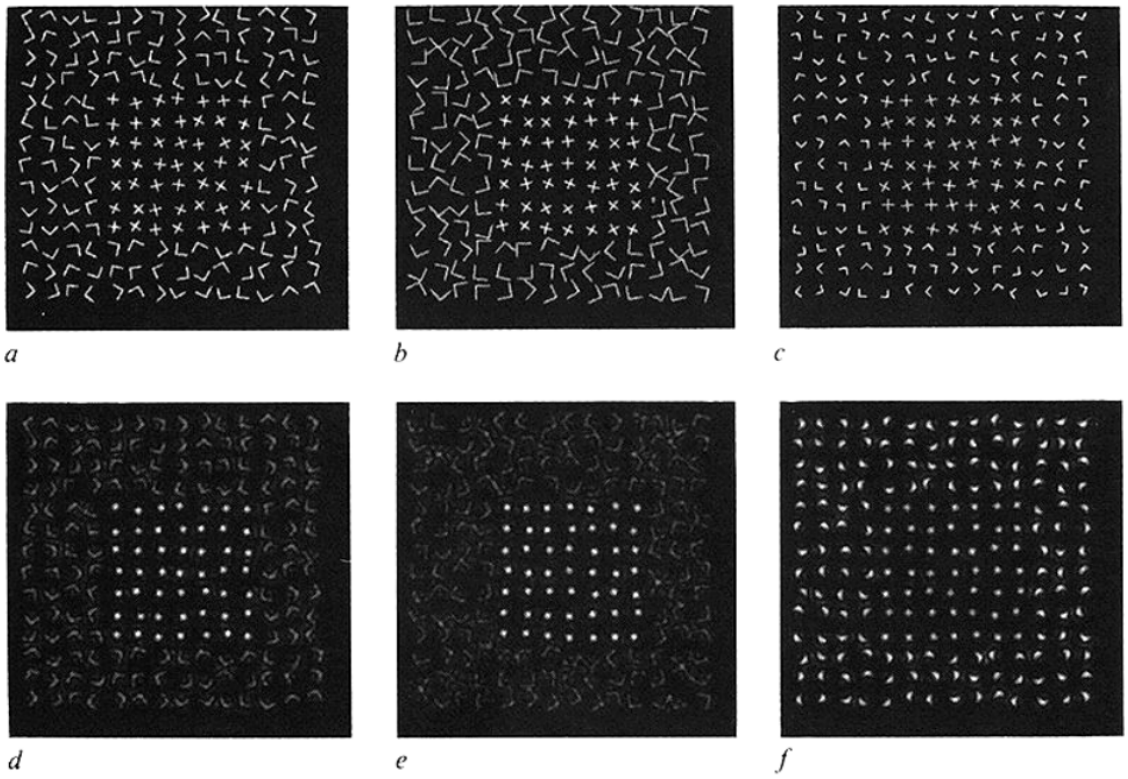


Figure 2.2: Results from [40]: The two textures shown in (a) get easier or harder to distinguish as the size of the "L" pattern increases in (b) or decreases in (c) even though the density of crossing and terminators has remained the same. However, this discriminability is predicted well by the responses of a centre surround filter shown in (d) - (f).

fairly good classification results were obtained.

From the mid nineties onwards, filter bank and wavelet based methods became increasingly successful at texture classification and synthesis. They were regarded as the method of choice. Their performance were improved because the richer representations of the filter response distributions. This is primarily due to two aspects. Firstly, full filter response distributions were learnt instead of only recording the low order moments. Secondly, the joint distribution or co-occurrence of filter responses were learnt rather than learning distributions for each filter independently. Another common trend was the raising of number of filters and wavelets to measure features at various scales and orientations.

During the period of mid nineties, synthesis and classification algorithms treated textures as if they are pure albedo patterns painted on a flat surface. Under this assumption, a single image could completely characterize all the possible variations of a texture patch. For example, synthetic affine transformations of a texture image were the reflections of what the texture looks like if it is physically rotated and scaled in the real world. However, it soon became apparent that such 2D texture models were not very physically plausible because they ignored all 3D effects including surface illumination changes, scale and perspective effects, normal variations, BRDF

variations and so on.

To fill the gap of 3D texture effects, in the late 1990s, the Columbia-Utrecht (CUReT) database [47] was introduced and included over 200 images of each of 61 materials taken under different viewpoints and illumination (see subsection 2.2.1). The database were initially used to demonstrate that 3D texture synthesis on a cylindrical surface gave much more realistic results as compared to traditional 2D texture mapping. However, the database later became an important dataset not only for synthesis but also for building and testing theoretical models of 3D textures.

Because 3D effects have a strong impact on the appearance of real world textures (see figure 2.3), the next phase in texture classification was to bring 3D textures within the ambit of the problem. The filter bank based methods were once again used.

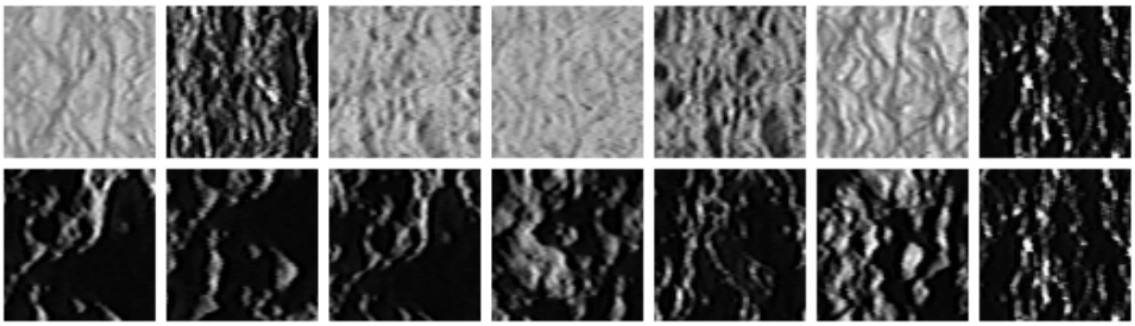


Figure 2.3: The change in imaged appearance of the same texture (Plaster B, texture from the CUReT database) with variation in imaging conditions. row 1): constant viewing angle and varying illumination. row 2): constant illumination and varying viewing angle. There is a considerable difference in the appearance across images.

By early 2000s, the problem of classifying 3D textures under varying viewpoint and illumination were proposed by Leung and Malik in [30]. Their works provided a decent innovation. It gave an operational definition of a texton based on filter responses and clustering. A 2D texton was defined as a cluster center in the filter response space. This generated textons automatically from not only an image but also a universal set of textons for a whole dataset. To compensate for 3D effects, 3D textons were proposed to be the cluster centers of filter responses over a stack of 20 images with representative viewpoints and lighting. The frequency distribution of these textons was sufficient for classifying a set of registered novel images taken under the same conditions. A nearest neighbor classifier based on the χ^2 distance was used. While the method was very important as it set up a framework in which 3D textures could be classified successfully, there were some limitations in a way that it needed many images for classification which had to be taken under exactly the same conditions as those used training.

At the same period, filter banks also had good performance in other closely related areas. In particular, the success of Bayesian classification applied to filter responses was convincingly demonstrated by Konishi *et al.* [48]. The joint probability distribution function (PDF) from responses of six filters were used to classify classes such as air, road, vegetation, and so on, and utilized them to label individual pixels

in the San Francisco outdoor datasets (The San Francisco database has 37 images of outdoor scenes taken on the streets of San Francisco. It has been segmented manually into 6 classes: Air, Building, Car, Road, Vegetation and Trunk). Therefore, their works were able to do both classification and segmentation depending solely on domain specific knowledge. There was also some preliminary investigation to see whether or not pre-trained PDFs on one dataset could be used to classify and segment the other. However, there was not enough attention paid on the variation of 3D textures, which changes considerably with imaging conditions, and only a single distribution of filter responses was learned for each class.

Similarly, Schmid *et al.* [49] modeled the joint PDF of thirteen rotationally invariant filters for the purposes of image retrieval. In addition, even the co-occurrence of these thirteen dimensional filter responses was modelled to achieve impressive results. Cula *et al.* [50] then addressed some of the major shortcomings of Leung and Malik's algorithm. They demonstrated that 2D textons (learnt from filter responses of single images instead of image stacks) could themselves be used for uncalibrated, single image classification without compromising on performance. The use of 2D textons allowed a texture to be characterized by multiple probability distributions (models). Theoretically as many as one from each training image could be learnt to sample the effects of viewpoint and illumination variations. In practice, only a few models were needed as the rest were discarded by a manifold shape preserving technique for model reduction. A somewhat similar approach was independently suggested in [31].

The problem of reducing the number of models required to characterize a texture is a major one and, generally speaking, two different approaches have been proposed.

The first approach is Geometric and focuses on building affine invariant texture descriptors in order to reduce the number of models needed to cope with variation in camera pose. Schaffalitzky *et al.* [51] exploited the fact that a texture with sufficient directional variation can be pose normalized by maximizing the weak isotropy of its second moment matrix (the technique is applicable in the absence of 3D texture effects). In essence, two images of the same texture which differ by an affine transformation are reduced to a canonical frame where they differ by only a similarity transformation. Full invariance can then be achieved by using a scale and rotation invariant filter bank to extract features. One drawback of this technique is that the proposed normalization is global rather than local. Not only would local normalization be more robust but it would also allow the method to be extended to textures which are not globally planar but which can be approximated as being locally planar. Realizing this, Lazebnik *et al.* [52] proposed an alternative method of generating local, affine invariant, texture features. In their framework, certain interest regions were first detected using a Laplacian blob detector. The characteristic scale at each point was determined and the region pose normalized locally. Spin images were then used instead of filter banks to generate affine invariant features for each region. The system achieved good classification results on both the Brodatz and the UIUC datasets.

In the second approach to model reduction, concepts from Machine Learning can be used to select a subset of the models while maximizing some criteria of

classification and generalization. A good example of this is work of Hayman *et al.* [53] where the nearest neighbor classifier used in [31] is replaced by a Support Vector Machine. They show that this not only improves classification performance on the CUReT database but also provides a principled way of selecting the required models. The average number of support vectors used is demonstrated to be 10 - 20% lower than the number of models required by a nearest neighbor classifier. The paper also considers how far pure learning approaches can go towards coping with imaging variations, especially those due to scale. At the time, Varma [54] figured out classification performance of Hayman's work was acceptable because the scaled images were included in the training set otherwise the results deteriorated very rapidly. He also found that a similar effect was observed for different instances of the same material, i.e. training on one instance of a material was no guarantee that another instance could be classified correctly.

To summarize, the texture classification problem has matured considerably over the last decades. The emphasis in the eighties was on separating patterns in synthetic binary images. This progressed in the early nineties to attempting classification of gray scale images of real world textures but with 2D variations due to synthetic rotations and scaling. Finally, in the late nineties, the classification task embraced real world 3D textures with real variations caused due to changing viewpoint and illumination. Throughout this period the most effective solution had been provided by filter banks which had themselves progressed by building more and more complex representations. They were introduced at first due to their biological plausibility and were soon seen as operators to extract features at multiple orientations and scales. Initially, during the early nineties, feature vectors were formed from only the mean and variance of filter response distributions. This has then changed considerably and the full joint PDF of filter responses is modeled. However, it must be pointed out that some recent papers still persist in attempting the 2D problem and approach classification via the dated technique of concatenating the low order moments of distributions to form feature vectors [55].

2.1.2 Optimal filtering

All the algorithms discussed so far had chosen their filter banks heuristically rather than by optimizing classification rates. It is therefore expected that the performance of the algorithms will get even better if their filters were to be replaced by the optimized ones, specifically for the given classification task. While there were attempts made at designing optimal filter banks they have not had much of an impact on the field in this period of time, the early 2000s. This is because such methods tend not to minimize the classification error itself but rather optimize other criteria, such as the separation between filter responses, in the hope that this will decrease the error. Such choices are necessary because it is often impossible to analytically express classification error as a function of the input filter bank while numerical techniques for classification error minimization are often computational expensive. Therefore, these optimization techniques were not widely used but were nevertheless important as tools to reason about the filtering process. We therefore present a very brief

overview of different optimization methods of the time in this subsection.

On one end of the spectrum are methods which determine filter banks and wavelets to be optimal since they are optimally localized in space and frequency [56], or because of their shift invariance and regularity properties [57] or because of the effective multiresolution representations of images using orthogonal wavelets [58]. Other relevant methods are optimal at characterizing textures though these are not optimal at discrimination. For instance, the 'eigenfilters' [59] are designed by obtaining eigenvectors of the corresponding variance-covariance matrix over a homogeneously textured region. Another method belongs to this category is the predictive linear filter [60] which is similar to a Gaussian Markov random field model [61] where the parameters are learnt using the pseudo-likelihood estimate.

Moving along the spectrum approach, there are also methods which choose the best subset of filters from a fixed filter bank. For instance, Zhu *et al.* [62] proposed starting with a large set of filters and then iteratively choosing an optimal subset which maximizes the L_1 norm of the probability distributions between classes. Similarly, parameters of Gabor filters were optimized for texture segmentation [63], these works tuned the filters which had been existed to push up performance.

A more general approach was taken by methods which applied techniques from Discriminant Analysis. Most such approaches focused on the two class problem and theorized that classification errors would decrease if the separation between the filter responses of the two classes had been maximized (due to the reduced PDF overlap). Various methods of measuring the distance between two classes was proposed in the literature. For example, i) the local linear transform method [64], is closely related to filter bank analysis methods and gives a statistical justification for the extraction of texture properties by means of convolution operators or local matches; or ii) using filters to discriminate textures [65] such as terrains, background surfaces, and random image fields. The optimum filter coefficients are determined by use of eigenvector analysis. The distance measurements were used as in (equation 2.1)

$$J_F = \frac{(\mu_1 - \mu_2)^2}{\sigma_1^2 + \sigma_2^2}; J_U = \frac{(\mu_1 - \mu_2)^2}{\mu_1^2 \mu_2^2}; J_{MS} = \frac{\mu_1}{\mu 2} \tag{2.1}$$

where μ_1, σ_1, μ_2 and σ_2 are the means and variances of the two classes respectively. The linear SVM also provides a measure of the maximum separation between two classes and relates it directly to classification. In fact, the normal to the separating hyperplane acts as an optimal filter. As such, a linear SVM should often turned out to be the best optimizer at the time, specially as its underlying assumptions are the least restrictive. However, a major drawback of such methods is that they do not generalize readily to many class problems. Even though an n class problem can be decomposed into $\mathcal{O}(n^2)$ two class problems, this results in a large number of filters for most classification applications at the time.

In the late 1990s, Jain *et al.* introduced neural network for texture classification [66]. Instead of using a general filter bank, a neural network is trained to find a minimal set of specific filters, so that both the feature extraction and classification tasks are performed by the same unified network. They noted that the first layer weights in a neural network essentially play the same role as a filter bank. Therefore, training a

neural network classifier to learn the weights is equivalent to designing optimal filters for the given classification task. Furthermore, LeCun *et al.*, the pioneers of Convolutional Neural Network (CNN), introduced CNN in [67] for document recognition. It is a multilayer neural networks trained with the back-propagation algorithm applying gradient based learning technique. CNNs, which are specifically designed to deal with the variability of 2D shapes, alongside deep learning technique, are shown to outperform all other techniques, and have been considered as the most popular methods despite of the high demand on resources (a lot of data needed to train the network, and computation cost) . In the recent days, hybrid CNN methods, such as FV-VGGVD [8][1] and a hybrid Scattering-ConvoNet [68], still have the highest performance on the domain of texture analysis.

To sum up, among those approaches, the LBP-family can be considered as a popular feature method which extracts well local micro-structure information from images. Ojala *et al.* first introduced the LBP method in 1996, then a multi-resolution version [69] in 2002. After that, several extensions on LBP have been conducted. In 2007, Tan *et al.* extended LBP to three-valued codes to become the local ternary pattern (LTP) [70]. Liao *et al.* proposed dominant LBP (DLBP) [71] which combines the most frequently occurred patterns with the Gabor filter responses for features. Later Guo *et al.* introduced completed LBP (CLBP) [24], which merges three components the sign (CLBP_S), magnitude (CLBP_M), and center pixel intensity (CLBP_C) together to form features. This enhances discriminative power compared to the original version. Variance in LBP (LBPV) [72] is used to encode local contrast information without requiring a quantization process, rotation invariance is implemented by estimating principal orientations and aligning LBP histograms. By constructing a cascading spatial pyramid of LBP, Qian *et al.* [73] introduced pyramid transformed LBP (PLBP), the robustness of PLBP was compared with those of other LBP variants in this works. Further, Liu *et al.* suggested extended LBP [74] by a combination of pixel intensities and local differences. In this way, the pixel intensity part is divided into a central pixel's component and neighbor's component. Likewise, the local difference consists of two components: radial differences and an angular difference. At the end, those four were combined to form features. In addition, Zhao *et al.* in [75] presented local binary pattern histogram Fourier features (LBP-HF) which implements rotation invariance by computing discrete Fourier transforms of LBP histograms. In [76], moreover, Guo *et al.* presented a three-layered learning framework in which LTP and CLBP were used as raw features to train and select the discriminative features.

Contrary to the micro-structure descriptors of LBP family, several broader range feature methods have been developed. Bovik *et al.* applied the Gabor filters to compute the average filter responses for features [77]. Mallat proposed the multiresolution wavelet decomposition method [58], which generates coefficients from the high-low (HL), low-high (LH), and low-low (LL) channels for subsequent classification tasks. Porter *et al.* [78] removed the high-high (HH) wavelet channels and

[1]FV-VGGVD is the **F**isher **V**ector aggregation of SIFT [11] features combined with those of a pretrained very deep convolutional network developed by **V**isual **G**eometry **G**roup from University of Oxford, VGGNet [12].

combined the LH and HL wavelet channels to obtain rotation invariance wavelet features. Haley *et al.* [79] calculated isotropic rotation invariance features from Gabor filter responses. More recent, scattering transform is considered as a high performance approach based on cascading wavelet transform layers [35] compared to previous wavelet-based methods.

2.2 Texture image datasets

We conclude this chapter by describing the four databases that have been used in this thesis. The first one is the Columbia-Utrecht (CUReT), it is a collection of 61 real-world surfaces, its samples were chosen to span a wide range of geometric and photometric properties. The second dataset is KTH-TIPS (Textures under varying Illumination, Pose and Scale) [80], KTH is the abbreviation of a university, this image database was created to extend the CUReT. The third dataset is University of Oulu Texture Database (OUTEX) [9] contains a collection of surface textures and natural scenes. The last presented dataset is University of Illinois at Urbana-Champaign (UIUC) texture database [52] features 25 texture classes. They are usually used to benchmark the performance of classification algorithms, and therefore used in the experiments of our proposed methods presented in Chapter 5, 6, and 7.

2.2.1 The Columbia-Utrecht (CUReT) database

The Columbia-Utrecht (CUReT) [47] is a texture data set of the visual appearance of real-world surfaces. It consists of in 3 databases:

1) BRDF (bidirectional reflectance distribution function) database with reflectance measurements for over 60 different samples, each observed with over 200 different combinations of viewing and illumination directions.

2)BRDF parameter database with fitting parameters from two recent BRDF models: the Oren-Nayar model [81] and the Koenderink *et al.* representation [82]. These BRDF parameters can be directly used for both image analysis and image synthesis.

3)BTF (bidirectional texture function) database with image textures from over 60 different samples, each observed with over 200 different combinations of viewing and illumination directions. Each of these databases is made publicly available for research purposes. For details about the measurements, and fitting procedures a technical report and summary paper are provided.

A collection of 61 real-world surfaces is used in the measurements. The samples were chosen to span a wide range of geometric and photometric properties. The categories include specular surfaces (aluminum foil, artificial grass), diffuse surfaces (plaster, concrete), isotropic surfaces (cork, leather, styrofoam), anisotropic surfaces (straw, corduroy, corn husk), surfaces with large height variations (crumpled paper, terrycloth, pebbles), surfaces with small height variations (sandpaper, quarry tile, brick), pastel surfaces (paper, cotton), colored surfaces (velvet, rug), natural surfaces (moss, lettuce, fur) and man-made surfaces (sponge, terrycloth, velvet). Different samples of the same type of surfaces are denoted by letters, e.g. Brick_a and Brick_b. Samples 29, 30, 31 and 32 are close-up views of samples 2, 11, 12 and

14, respectively.

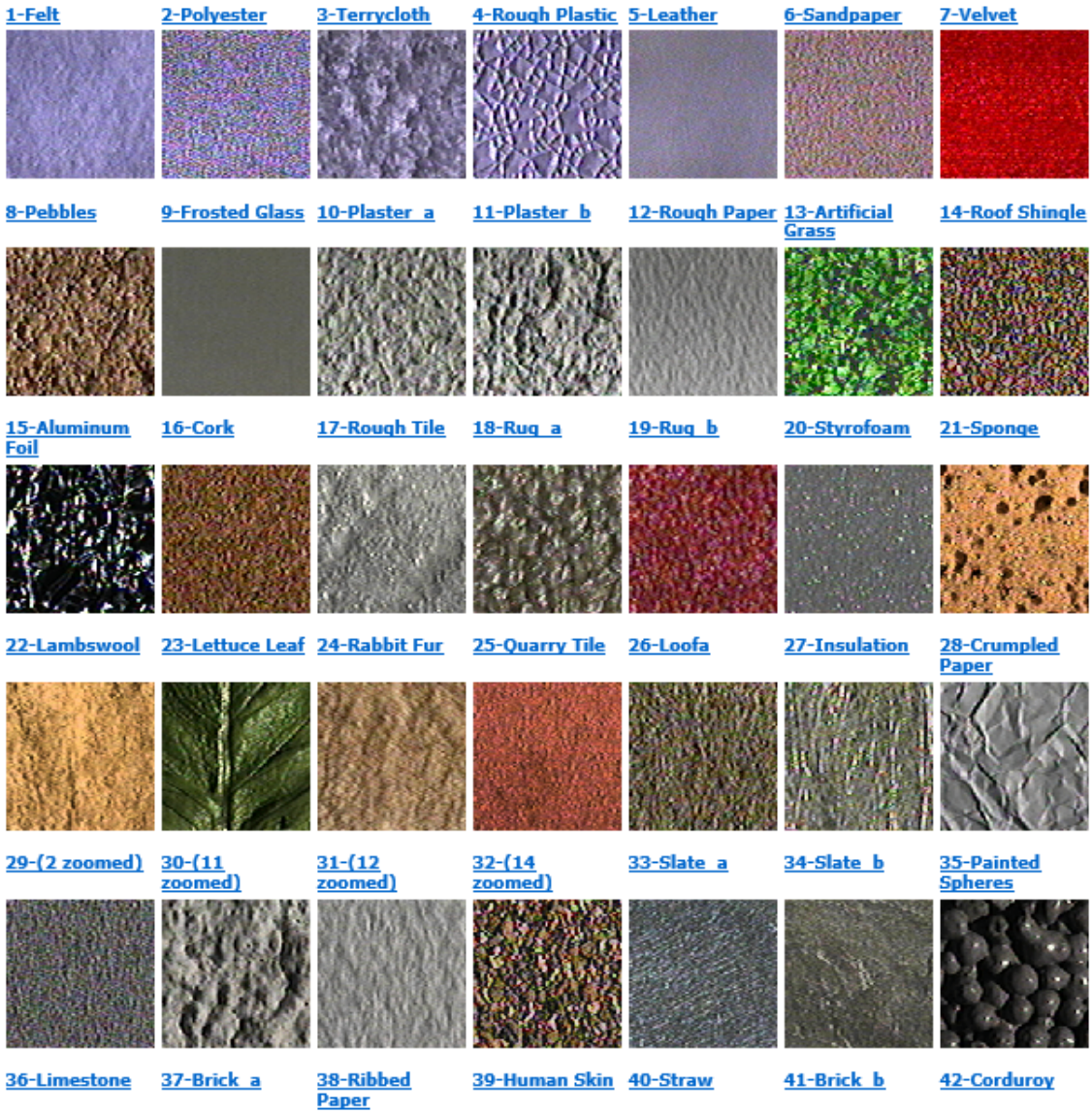


Figure 2.4: CUReT samples

2.2.2 KTH-TIPS and KTH-TIPS2

The KTH-TIPS (Textures under varying Illumination, Pose and Scale) [80], KTH is the abbreviation of a university, image database was created to extend the CUReT database in two directions, by providing variations in scale as well as pose and illumination, and by imaging other samples of a subset of its materials in different settings.

While the CUReT database[47] images 61 materials, the KTH-TIPS database currently contains images of 10 of those materials as outlined in Table 2.1 and depicted in Figure 2.5. Each of the samples is planar. The Orange peel was flattened by placing it inside a CD case.

Table 2.1: The materials present in the KTH-TIPS database.

CUReT	Corresponding CUReT sample number
Sandpaper	06
Crumpled aluminium foil	15
Styrofoam	20
Sponge	21
Corduroy	42
Linen	44
Cotton	46
Brown bread	48
Orange peel	55
Cracker B	60

Figure 2.5: Images of the materials present in the KTH-TIPS database.

The KTH-TIPS2 databases [83] took this a step further by imaging 4 different samples of 11 materials, each under varying pose, illumination and scale.

There are two versions of KTH-TIPS2. KTH-TIPS2-a does not contain an equal number of images for each sample (40 out of 44 samples have 108 images, but 4 samples contain only 72 images). KTH-TIPS2-b contains also these "missing" images.

The acquisition of KTH-TIPS2 images largely followed the procedure used for KTH-TIPS, though with some differences with regard to scale and illumination. The images were taken with an Olympus C-3030ZOOM digital camera at a resolution of 1280×960 pixels. Many of the full-size images contain not only the sample, but also some background.

Like KTH-TIPS, KTH-TIPS2 contains images at 9 scales equally spaced logarithmically over two octaves. However, in KTH-TIPS2 the scale closest to the camera corresponds to Scale 2 of KTH-TIPS. This is due to problems with focus in Scale 1

of KTH-TIPS. The scales used are described in full in Table 2.3, and full-resolution images from one material (Cracker B) are shown in Figure 2.6. To maintain compatibility with KTH-TIPS, these scales are labeled with numbers ranging from 2 to 10 rather than 1 to 9. In other words, scale 4 in KTH-TIPS and scale 4 in KTH-TIPS2 represents the same camera-object distance.

KTH-TIPS2 contains images at the same 3 poses as KTH-TIPS (frontal, rotated 22.5° left and 22.5° right), but 4 rather than 3 illumination conditions. The 3 illuminations from KTH-TIPS are used (frontal, 45° from the top and 45° from the side, all taken with a desk-lamp with a Tungsten light bulb), and for the fourth illumination condition we switched on the fluorescent lights in the laboratory.

At each scale 12 images were taken in a combination of three poses (frontal, rotated 22.5° left and rotated 22.5° right) and four illumination conditions (from the front, from the side at roughly 45° and from the top at roughly 45°, and using ambient lighting), sample images are presented in Figure 2.6. Note that images 1-9 follow the naming convention of KTH-TIPS, whereas images 10-12 are the images with the new, ambient, lighting condition. This gives a total of $12 \times 9 = 108$ images per sample. However: For "Aluminium foil", "Linen" "Cotton" and "Cracker", i.e. four of the six classes already in KTH-TIPS (Table 2.2), images were acquired from 3 new samples, and inserted the original KTH-TIPS images into the KTH-TIPS2 database as the fourth sample with the caveat that those four samples were not captured with ambient lighting, and only 8 scales (2-9) were available.

The 200×200 pixel patches (cropped to remove the background) may be obtained as color PNG images for KTH-TIPS2-a (374MB) and KTH-TIPS2-b (386MB).

Table 2.2: The materials present in the KTH-TIPS2 database.

Material	Corresponding CUReT sample number	Present in KTH-TIPS
Crumpled aluminium foil	15	×
Cork	16	
Wool	22	
Lettuce leaf	23	
Corduroy	42	×
Linen	44	×
Cotton	46	×
Brown bread	48	×
White bread	52	
Wood	54	
Cracker	59 and 60	×

2.2.3 OUTEX database

OUTEX database [9], University of **OU**LU **Tex**ure database, contains a collection of surface textures and natural scenes (at an early stage). The collection of surface

Table 2.3: The materials present in the KTH-TIPS2 database.

Scale number	Relative scale	Distance to camera (cm)
2	$2^{-1.00} = 0.500$	16.65
3	$2^{-0.75} = 0.595$	19.80
4	$2^{-0.50} = 0.707$	23.55
5	$2^{-0.25} = 0.841$	28.00
6	$2^{0.00} = 1.00$	33.30
7	$2^{+0.25} = 1.189$	39.60
8	$2^{+0.50} = 1.414$	47.09
9	$2^{+0.75} = 1.682$	56.00
10	$2^{+1.00} = 2.000$	64.41

textures is regularly expanded. Recently, the database contains 320 surface textures, both macro- and micro-textures.

Many textures in OUTEX have variations in local color content, which results in challenging local gray scale variations in intensity images. Some textures have a large tactile dimension, which can induce considerable local gray scale distortions. The current database of surface textures comprises of 27054 images, in both 24-bit RGB and 8-bit gray scale, totaling about 40 GB of disk space.

Each texture uses three different simulated illuminants provided in the light source: 2300K horizon sunlight denoted as 'horizon', 2856K incandescent CIE A denoted as 'inca', and 4000K fluorescent TL84 denoted as 'TL84'. Every texture is captured using six spatial resolutions (100, 120, 300, 360, 500 and 600 dpi) and nine rotation angles ($0^\circ, 5^\circ, 10^\circ, 15^\circ, 30^\circ, 45^\circ, 60^\circ, 75^\circ$ and 90°) .

With three illuminants, six spatial resolutions and nine rotation angles, there are 162 RGB images captured for every texture sample. Eight bit intensity images are generated from the RGB images using the standard formula:

$I = 0.299R + 0.587G + 0.114B$.

OUTEX contains three basic types of test suites: texture classification ("TC"), supervised texture segmentation ("SS"), and unsupervised texture segmentation ("US"). In this thesis, we mention only the classification test suites.

The purpose of an individual test suite is to encapsulate a meaningful entity for the purpose of empirical evaluation of a candidate texture analysis algorithm. A test suite may contain a large number of classification or segmentation problems having a particular basic structure, but for example different partitioning of the image data into training and testing sets due to randomization, or different collection of textures. The motivation in packing a large number of problems with well defined variation for example in terms of textures is to properly evaluate the robustness of the algorithm with respect to its built-in parameters (any external parameters provided by the user are required to remain constant throughout the suite) and analyzed textures. Further, individual test suites may be combined into challenging "grand suites" assessing the performance in many different respects. The texture samples are illustrated in Figure 2.7.

Figure 2.6: Images of the materials present in the KTH-TIPS2 database.

2.2.4 UIUC database

The UIUC texture database [52] features 25 texture classes, 40 samples each, and includes changes of viewing angle, scale, and illumination conditions. All images are

Table 2.4: OUTEX Database: Surface categories and the number of textures.

Categories	Number of Textures
barleyrice	11
canvas	46
cardboard	1
carpet	12
chips	23
crushedstone	8
flakes	10
flour	13
foam	4
fur	12
granite	10
granular	3
gravel	7
groats	7

Categories	Number of Textures
leather	5
mineral	6
paper	10
pasta	6
pellet	4
plastic	47
quartz	6
rubber	1
sand	5
sandpaper	8
seeds	13
tile	7
wallpaper	20
wood	12
wool	2

in grayscale JPG format, 640×480 pixels.

The database consists of surfaces whose texture is due mainly to albedo variations (e.g., wood and marble), 3D shape (e.g., gravel and fur), and a mixture of both (e.g., carpet and brick). There are significant viewpoint changes and scale differences within each class, and the illumination conditions are uncontrolled. Additional sources of variability is added wherever possible during data acquisition. These include non-planarity of the textured surface (bark), significant non-rigid deformations between different samples of the same class (fur, fabric, and water), inhomogeneities of the texture pattern (bark, wood, and marble), and viewpoint-dependent appearance variations (glass). Figure 2.8 illustrates four samples of 25 texture classes in the database.

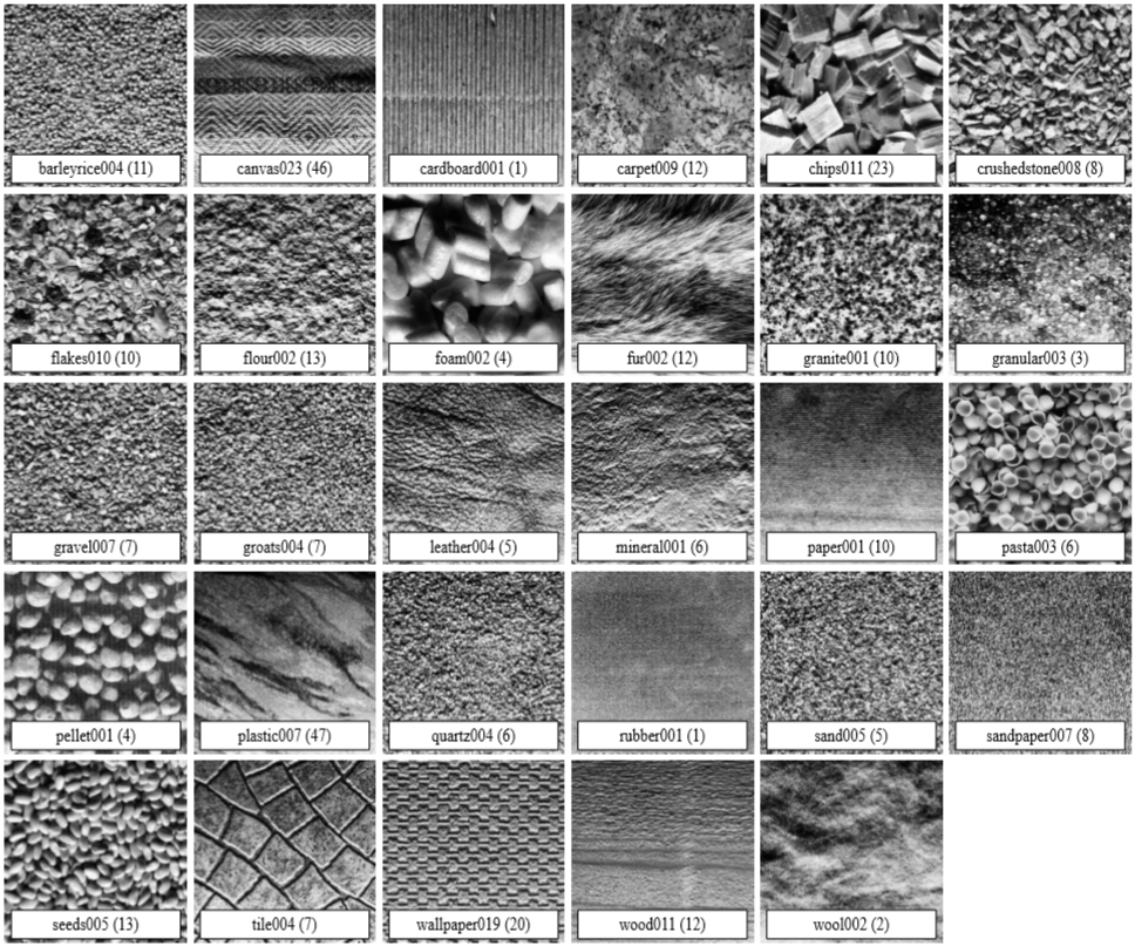


Figure 2.7: OUTEX texture dataset: One example from each category. The number in parentheses denotes the number of textures in that category. The images are 512×512 pixels in size printed at 464 dpi and histogram equalized for visualization purposes.

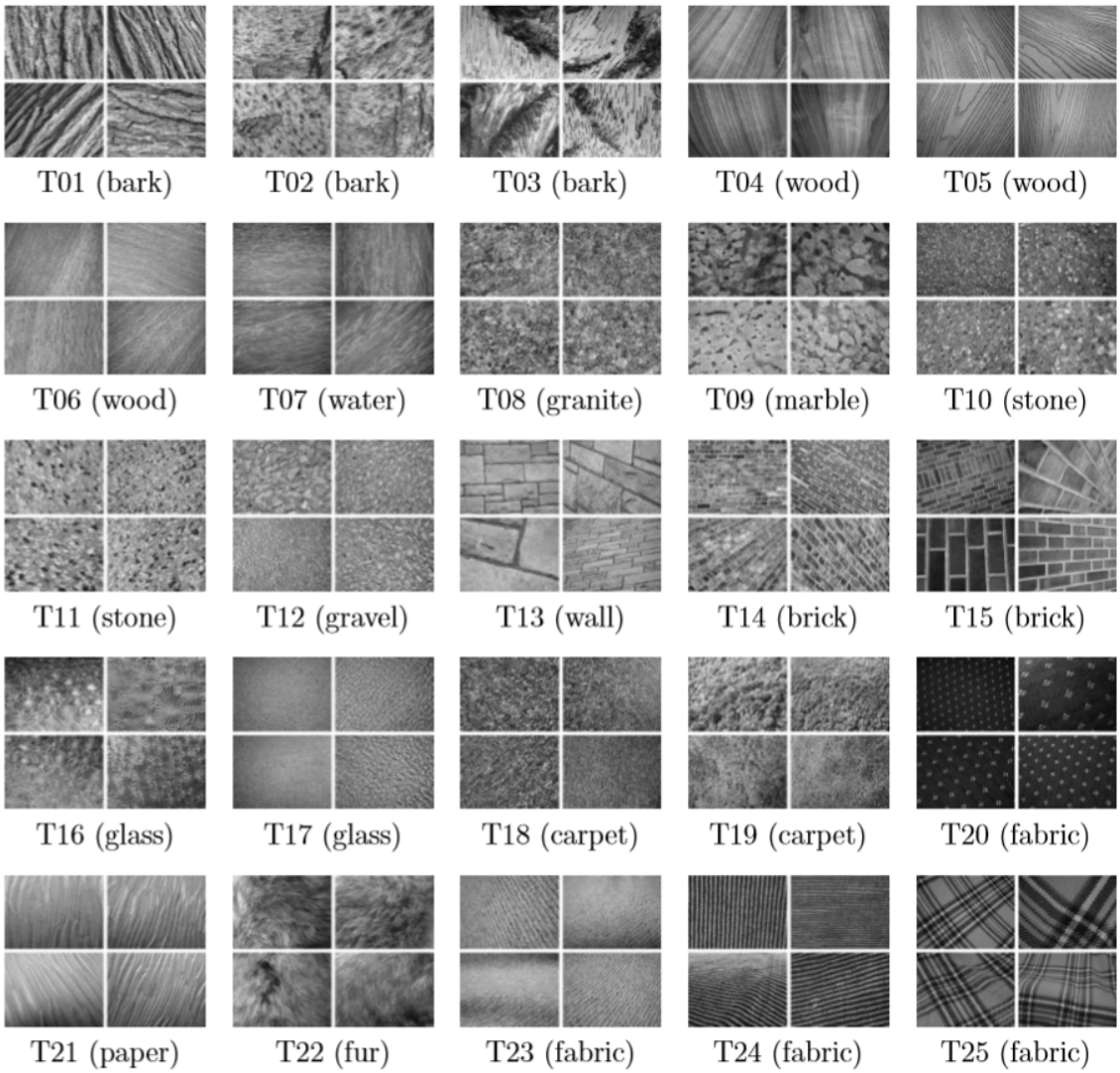


Figure 2.8: UIUC texture dataset: Four samples each of the 25 texture classes.

Chapter 3

Local Binary Pattern

The local binary pattern (LBP) is an image operator which encodes an image into an array or image of integer labels describing small-scale appearance of the image. These labels or their statistics, most commonly the histogram, are then used for further image analysis. The most widely used versions of the operator are designed for monochrome still images but it has been extended also for color (multi channel) images as well as videos and volumetric data. This chapter covers the different versions of the actual LBP operator in spatial domain which is commonly used for texture classification and close-related applications.

3.1 Original LBP

The original LBP was firstly proposed in 1994 and then appeared again in a comparative study in 1996. The introduction of the $LBP^{ri}_{r,p}, LBP^{u2}_{r,p}$ and, $LBP^{riu2}_{r,p}$ descriptors [29] was in 2002, since then LBP began to attract broad interest of the research community. The original LBP characterizes the spatial structure of a local image texture pattern by thresholding a 3×3 square neighborhood with the value of the center pixel and considering only the sign information to form a local binary pattern. As illustrated in Figure 3.1, given an image pixel c, the LBP pattern at c is computed by comparing the pixel's gray value g_c with the gray values g_c of its p neighbors $g_0, ..., g_{p-1}$ which are distributed in an equal angle on a circle of radius r centered at c:

$$LBP_{r,p(c)} = \sum_{i=0}^{p-1} S(g_i - g_c)2^i, \qquad S(x) = \begin{cases} 1 & x \geqslant 0 \\ 0 & x < 0 \end{cases} \tag{3.1}$$

In practice the neighboring pixels are sampled on a circle, such that the gray values of neighbors which do not fall exactly in the center of pixels are estimated by interpolation. The co-occurrence of the comparison results is recorded in $LBP_{r,p}(c)$, by a unique string of binary numbers, where the sign function $s()$ ensures that the LBP code is invariant against any monotonic transformation of image brightness.

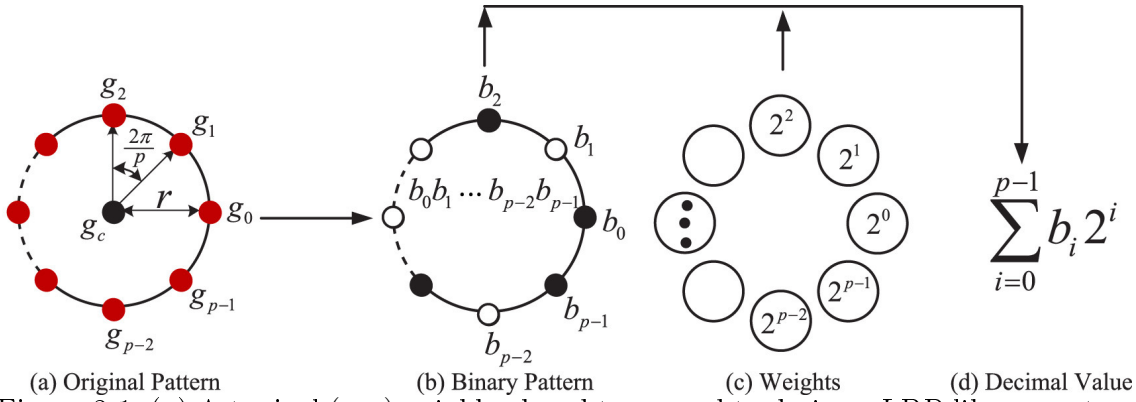


Figure 3.1: (a) A typical (r, p) neighborhood type used to derive a LBP like operator: central pixel g_c and its p circularly and evenly spaced neighbors $g_0, ..., g_{p-1}$ on a circle of radius r [10].

Given a texture image $f(x, y)$, it can be characterized by the distribution of LBP patterns, representing a whole image by a LBP histogram vector h:

$$h_i = \sum_{x,y} I\{f(x, y) = i\}, \quad i = 0, ..., n-1 \tag{3.2}$$

where n is the number of LBP patterns, and

$$I(A) = \begin{cases} 1 & , if \quad A \quad is \quad true \\ 0 & otherwise \end{cases} \tag{3.3}$$

By altering the radius r and the number of neighbors p, one can compute LBP features for any quantization of the angular space and for any spatial resolution. LBP has several distinctive advantages, such that it is not only easily to implement but also is invariance to monotonic illumination changes, and low computational complexity. Despite these merits, there are significant disadvantages associated with the original LBP such as: (1) Producing rather long histograms, overwhelmingly large even for small neighborhoods, leading to decreased distinctiveness and large storage requirements. (2) Capturing only the very local structure of the texture and failing to detect large-scale textural structures. (3) Being sensitive to image rotation. (4) Being highly sensitive to noise: the slightest fluctuation above or below the value of the central pixel is treated the same way as a major contrast. (5) Losing local textural information due to the use of hard, fixed and coarse quantization and only the signs of differences of neighboring pixels are utilized.

To address these limitations, the traditional fundamental LBP strategy led to three early generalizations:

3.1.1 Rotation invariant LBP

The original LBP descriptor is not rotationally invariant, a serious limitation for many real world applications. A rotation invariant version $LBP^{ri}_{r,p}$, of $LBP_{r,p}$, was obtained by grouping together those LBPs that are actually rotated versions of the

same pattern, introduced by Pietikäinen *et al.* in [84]. Formally, the $LBP^{ri}_{r,p}$ is defined as

$$LBP^{ri}_{r,p} = \min\{ROR(LBP_{r,p}, i) \quad |i = 0, 1, ..., p-1\} \tag{3.4}$$

where function $ROR(x, i)$ performs a circular i-step bit-wise right shift on the pattern binary string x, i times. Keeping only those rotationally-unique patterns leads to a significant reduction in feature dimensionality, however the number of codes in $LBP_{r,p}$ still increases rapidly with p.

3.1.2 Uniform LBP

Ojala *et al.* in [69] observed that some LBP patterns occur more frequently than others, therefore the uniform LBP $LBP^{u2}_{r,p}$ preserves only the uniform patterns and groups all information contained in the nonuniform patterns. In particular, the uniformity measure

$$U(LBP_{r,p}) = \sum_{i=1}^{P} |s(g_{mod(i,p)} - g_c) - s(g_{i-1} - g_c)| \tag{3.5}$$

counts the number of $0 \to 1$ or $1 \to 0$ transitions between adjacent bits in the binary string of the LBP code. All patterns with $U > 2$ are called nonuniform patterns and are classified under a single group, thus the 2^p original $LBP_{r,p}$ patterns are classified into $p(p-1)+3$ different groups, resulting in a significant dimensionality reduction.

3.1.3 Rotation invariant uniform LBP

In order to obtain improved rotation invariance and to further reduce the feature dimensionality, building on $LBP^{ri}_{r,p}$ and $LBP^{u2}_{r,p}$, Ojala *et al.* [69] proposed the rotation invariant uniform LBP descriptor

$$U(LBP^{riu2}_{r,p}) = \begin{cases} \sum_{i=0}^{p-1}(g_i - g_c) & ,if \quad U(LBP_{r,p}) \leqslant 2 \\ p+1 & otherwise \end{cases} \tag{3.6}$$

where the uniformity measure U is defined in (3.5). $LBP^{riu2}_{r,p}$ classifies all 2^p LBPs into $p+2$ distinct groups resulting in a significantly lower feature dimensionality.

$LBP^{ri}_{r,p}$ has all the shortcomings of the original LBP, except for having some level of rotation invariance. Unfortunately, the $LBP^{ri}_{r,p}$ descriptor was found to give poor performance for rotation invariant texture classification [69, 84].

In comparison with $LBP_{r,p}$ and $LBP^{ri}_{r,p}$, $LBP^{riu2}_{r,p}$ has much lower feature dimensionality. Although uniform patterns are beneficial in practice, their usage is still heuristic, i.e. the experimental observation that the uniform patterns appear to be fundamental properties of local image textures and have dominating proportions. This was challenged by researchers [33, 71, 85], who argued that uniform LBPs extracted from some images containing high curvature edges, crossing boundaries or corners are not necessarily the dominant patterns.

Many extensions and modifications of LBP will be discussed in the following sections, build on the fundamental LBP strategy, seeking to increase robustness, to

improve discriminative power, and to avoid the disadvantages of the traditional LBP methods.

3.1.4 Multiscale LBP

A significant limitation of the original LBP operator is its small spatial support area. Features calculated in a local 3×3 neighborhood cannot capture large-scale structures that may be the dominant features of some textures. However, adjacent LBP codes are not totally independent of each other. Figure 3.2 displays three adjacent four-bit LBP codes [86]. Assuming that the first bit in the leftmost code is zero, the third bit in the code to the right of it must be one. Similarly, the first bit in the code in the center and the third bit of the rightmost one must be either different or both equal to one. The right half of the figure shows an impossible combination of the codes. Each LBP code thus limits the set of possible codes adjacent to it, making the "effective area" of a single code actually slightly larger than 3×3 pixels. Nevertheless, the operator is not very robust against local changes in the texture, caused, for example, by varying viewpoints or illumination directions. An operator with a larger spatial support area is therefore often needed.

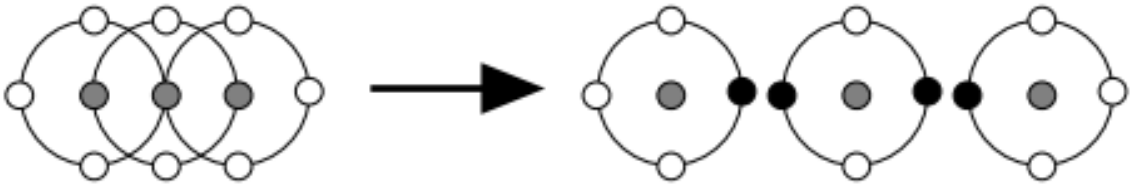


Figure 3.2: Three adjacent $LBP_{4,R}$ neighborhoods are impossible to combine LBP codes. A black disk denotes the gray level of a sample is lower than that of the center

A straightforward way of enlarging the spatial support area is to combine the information provided by N LBP operators with different P and R values. This way, each pixel in an image gets N different LBP codes. The most accurate information would be obtained by using the joint distribution of these codes. However, such a distribution would be overwhelmingly sparse with any reasonable image size. For example, the joint distribution of $LBP_{8,1}$, $LBP^{u2}_{16,3}$, and $LBP^{u}_{24,5}2$ would contain $256 \times 243 \times 555 \approx 3.5 \times 10^7$ bins. Therefore, only the marginal distributions of the different operators are considered even though the statistical independence of the outputs of the different LBP operators at a pixel cannot be warranted. For example, a feature histogram obtained by concatenating histograms produced by rotation-invariant uniform pattern operators at three scales (1, 3 and 5) is denoted as: $LBP^{riu2}_{8,1+16,3+24,5}$.

The aggregate dissimilarity between a sample and a model can be calculated as a sum of the dissimilarities between the marginal distributions

$$L_N = \sum_{n=1}^{N} L(S^n, M^n) \tag{3.7}$$

where S^n and M^n correspond to the sample and model distributions extracted by the nth operator [69]. The chi square distance or histogram intersection can also be

a substitution for the log-likelihood measure. Although the LBP codes at various radii are not statistically independent in the typical case, using multi-resolution analysis often enhances the discriminative power of the resulting features. With most applications, this straightforward way of building a multi-scale LBP operator has resulted in good accuracy.

3.2 Relation of LBP to earlier texture methods

In this section, LBP is presented as texture descriptor in the relationship with some others. Although LBP was not directly derived from any of the methods presented, it is very closely related to some of them. Those are co-occurrence matrix, N-tuple methods, and texton statistics. The LBP is related to many other well-known

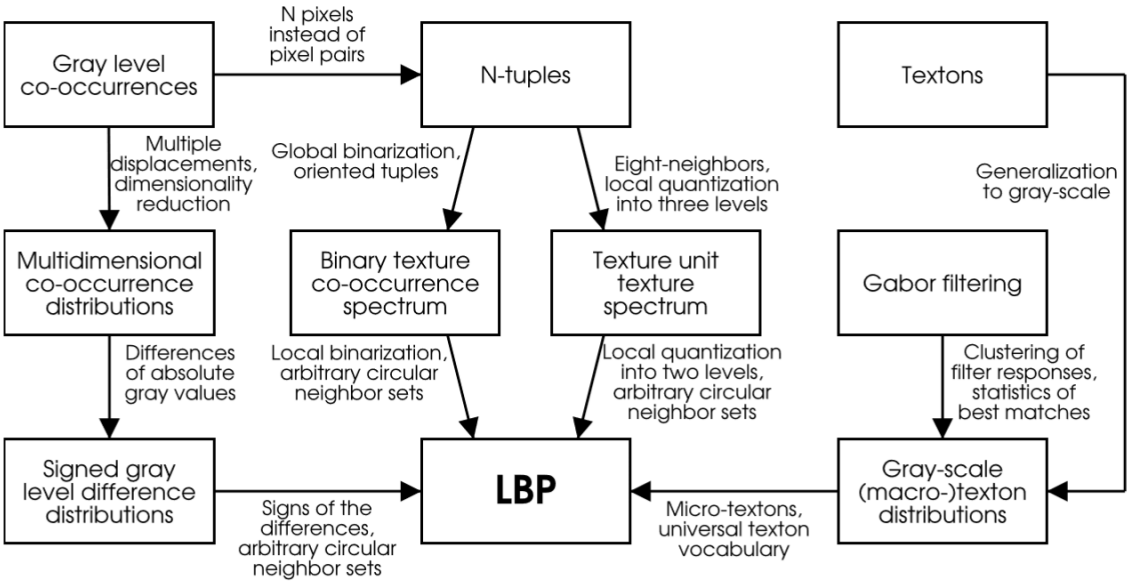


Figure 3.3: Relation of LBP to earlier texture methods

texture analysis features as presented in Figure 3.3 [87]. The arrows represent the relationship between different methods, and the texts next to the arrows summarize the main differences of those.

The original LBP

The LBP operator was first introduced as a complementary measure for local image contrast [23]. The original LBP works with the eight-pixel-neighbor, the gray level of the center pixel used as a threshold. An LBP code for a neighborhood is produced by multiplying the thresholded values with weights given to the corresponding pixels, and summing up the result (Figure 3.4). Since the LBP is, by definition, invariant to monotonic changes in gray scale, it is supplemented by an orthogonal measure of local contrast. Figure 3.4 shows how to derive the contrast measure (C). The mean value of the gray levels below the center pixel is subtracted from that of the gray levels above (or equal to) the center pixel. Two-dimensional distributions of the LBP and local contrast measures are used as features. The operator is called LBP/C, and good discrimination accuracy were reported in [23] with textures selected from the photographic album of Brodatz.

Cooccurrences and graylevel differences

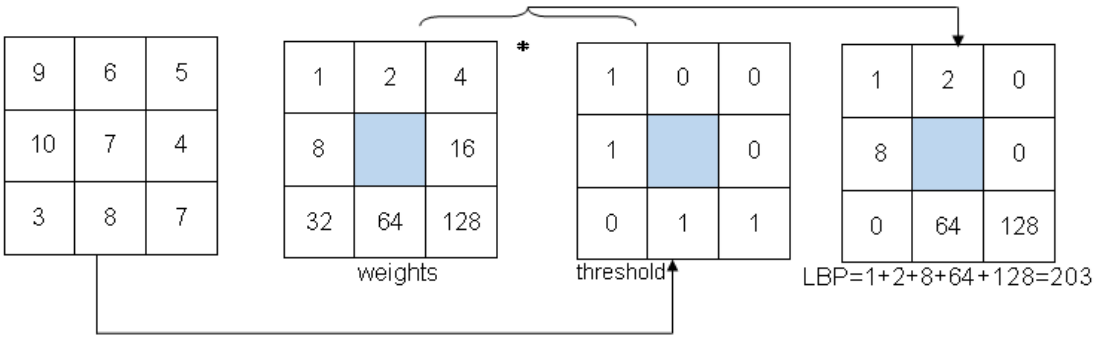


Figure 3.4: Calculating the original LBP code and a contrast measure

The graylevel cooccurrence(GLC) statistics were first described by Haralick *et al.* in 1973. Typically, GLC features are extracted in two stages.

1) A set of gray-level co-occurrence matrices (GLCM) is computed. This is done by selecting a few displacement operators, with which the image is scanned. Some typical sets of displacements are shown in Figure 3.5. While processing an image, all the pairs of pixels are found and positioned relative to each other as the displacement operator indicates. A statistic of the gray levels of these pixel pairs is collected into a two dimensional co-occurrence matrix. The number of rows and columns in this matrix equals the number of gray levels in the image. It means that the co-occurrence matrix represents the joint probability of the pairs of gray levels which occur at pairs of points separated by the displacement operator proposed in [88]. Since the co-occurrence matrix collects information about pixel pairs instead of single pixels, it is called a second-order statistic.

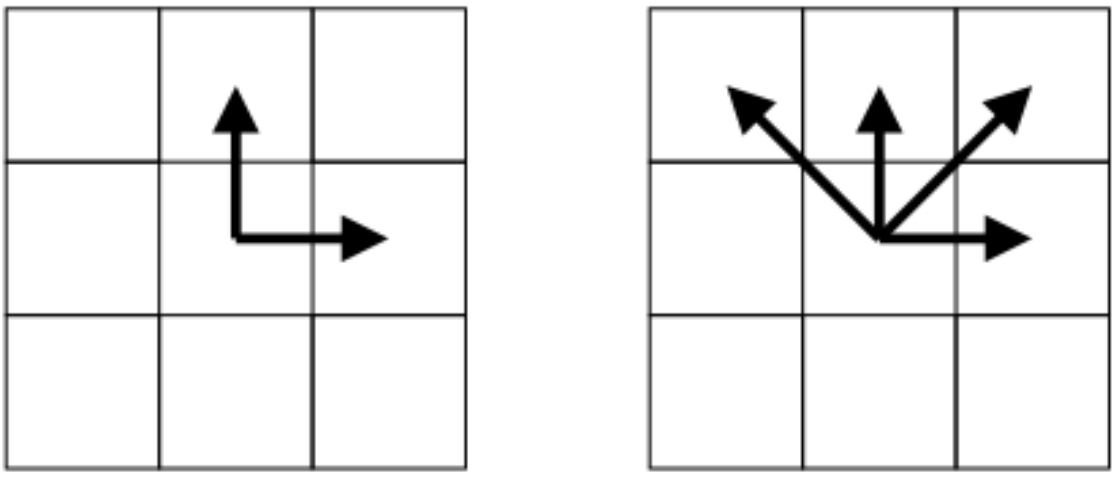


Figure 3.5: Displacement operators for the calculation of co-occurrence matrices.

2) In the second stage, actual features are extracted from an averaged co-occurrence matrix, or by averaging the features calculated from different matrices. The co-occurrence matrices give information regarding the homogeneity of an image, its contrast, and linearity, etc. [88] or choosing five efficient measures [89] extracted from the GLCMs: energy, entropy, correlation, homogeneity and inertia.

The gray-level difference (GLD) methods closely resemble the Weszka's co-occurrence

approach [27]. The main difference is that instead of the absolute gray levels of the pair of pixels, their difference is utilized. By this way it is possible to achieve invariance against changes in the overall luminance of an image. Another advantage is that in natural textures, the differences tend to have a smaller variability to the absolute gray levels, resulting in more compact distributions. As features, Weszka *et al.* [27] proposed to use the mean difference, the entropy of differences, a contrast measure, and an angular second moment.

Valkealahti *et al.* [90] presented a method in which multi-dimensional co-occurrence distributions were utilized. In their approach, the gray levels of a local neighborhood are collected into a high-dimensional distribution, whose dimensionality is reduced with vector quantization. For example, if the eight-neighbors are used, a nine-dimensional (eight neighbors and the pixel itself) distribution of gray values is created instead of eight two-dimensional co-occurrence matrices. An enhanced version of this method was later utilized by Ojala *et al.* [91] with signed gray-level differences.

According to the derivation of LBP [87], it is apparent that LBP can be regarded as a specialization of the multi-dimensional signed gray-level difference method. The reduction of dimensionality is directly achieved by considering only the signs of the differences, which makes the use of vector quantization unnecessary. A comprehensive comparison of the original LBP, the LBP/C, GLCM, GLD, and many other methods has been carried out by Ojala *et al.* [23].

Texture spectrum and N-tuple methods

At the turn of the 1990s, He & Wang introduced a new model of texture analysis based on the so-called texture unit (TU) [92, 93]. In their model, texture information was collected from a 3×3 neighborhood, which was divided into three levels according to the value of the center pixel. Each neighbor got the label 0, 1, or 2 depending on whether it was below, equal to or above the gray level of the center pixel, respectively. This resulted in a total number of $3^8 = 6561$ different texture units which were collected into a feature distribution called texture spectrum (TS). The method is called "TUTS" in the literature. The TUTS method [92] closely resembles the original LBP. The only difference is that TUTS uses three levels of thresholding whereas LBP uses two levels.

The N-tuple method was firstly described in [94], it is closely resemble the TUTS method. The main difference is that TUTS considers the eight-neighbors of a pixel whereas N-tuples consider N arbitrary samples. Patel *et al.* [95] first used oriented N-tuple operators which globally thresholded binary images using one-layer neural network. The method was named the binary texture co-occurrence spectrum (BTCS). The BTCS method was soon extended to gray-scale images in [96], resulted in a gray level texture co-occurrence spectrum (GLTCS). The rank coding was used as a measure to reduce the dimensionality of the features. Later, Hepplewhite & Stonham [97] returned to the binary representation by using the BTCS with binarized edge images. This method was called the zero crossings texture co-occurrence spectrum (ZCTCS).

The concept of arbitrarily sampled "neighborhoods" in the N-tuple methods can be considered as a precursor of the circular sampling in LBP. Furthermore, the binary

N-tuples in the BTSC method closely resemble LBP codes. The main difference is in the binarization, which is done locally in the LBP. Unfortunately, there has been no reports found which compares the LBP and the N-tuple methods.

Texton statistics

Textons were originally used by Julesz in [5] as the fundamental units of human preattentive texture discrimination. According to that research, textures consisted of separated binary texture primitives — orientation elements, crossings and terminators. Until recently, this model was not extended to gray-scale textures. Malik *et al.* [30] mentioned again the term "texton" in a study of texture segmentation. In their model, multi-dimensional features are created by filtering with a Gabor filter bank, and representative vectors (i.e. textons) are formed by k-means clustering. The texton vocabulary is created once for a set of textures, and used in describing all textures in it. Distributions of textons are used for texture description. Later, a number of works [30, 50, 98] have used similar models for recognizing textures in the CUReT database [47].

Varma *et al.* [98] considered the Gabor-based textons as micro-primitives. However, even the smallest Gabor filters calculate the weighted mean of pixel values over a small neighborhood. LBP in turn considers each pixel in the neighborhood separately, thus providing even more fine-grained information. LBP can therefore be regarded as a micro-texton, as opposed to the Gabor-based macro-textons. Another difference is that in LBP there is no need to create an application specific texton vocabulary. Instead, a generic vocabulary of micro-textons can be used for most purposes. The "Gabor texton" approach was recently compared to the LBP by Pietikäinen *et al.* [99] with 3-D textures from the CUReT database. The LBP provided better performance even with a more challenging set of textures and with a significantly smaller computational overhead. Later, Sánchez-Yáñez *et al.* [100] introduced a method called "coordinated clusters representation" (CCR) for texton-based texture discrimination. The CCR method extracts 3×3 neighborhoods from a globally binarized image, and uses the distribution of these local "textons" as a texture feature. The idea is very similar to the LBP with the difference of global binarization.

3.3 LBP variants

The success of LBP methods in different computer vision problems and applications has inspired many researchers to study for various variants. Due to its simplicity the LBP method can be easily modified to make it suitable for the needs of different types of problems. The original LBP has also some shortcomings that need to be eliminated. Therefore, several extensions and modifications of LBP have been proposed with an aim to increase its robustness and discriminative power. In this section different variants are divided into such categories that describe their roles in feature extraction. Some of the variants could belong to more than one category, but in such cases only the most obvious category was chosen. A summary of the variants is presented hereafter, and (Figure 3.6) illustrates a general feature extraction pipeline using LBP, where the actual sequence of stages taken clearly depends on the particular LBP method. The choice of a proper method for a given

application depends on many factors, such as the discriminative power, computational efficiency, robustness to illumination and other variations, and the imaging system used. Therefore the LBP (and LBP with contrast) operators presented in the previous sections will usually provide a very good starting point when trying to find the optimal variant for a given application.

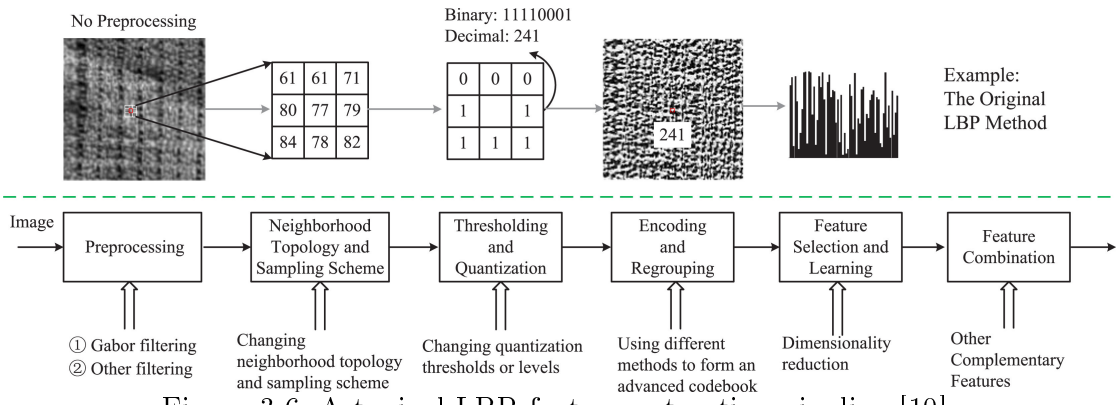


Figure 3.6: A typical LBP feature extraction pipeline [10].

3.3.1 Preprocessing

Preprocessing the input image before extracting LBP features is useful in many applications. It is commonly agreed that LBP captures small and fine details while Gabor filters encode appearance information over a broader range of scales. Thus, it is believed that Gabor filtering and LBP provide complementary information. For example, Local Gabor Binary Patterns (LGBP) [101] extracts LBP features from images obtained by filtering an image with 40 Gabor filters of different scales and orientations. This is a high performance method, its shortcoming is the high dimensionality of the LGBP representation. Vu *at al.* [102] developed an effective preprocessing technique. It is composed of difference of Gaussian (DoG) filtering, and thresholding to split a input image into two maps. LBP codes are then computed on these maps.

In some studies, edge detection has been used prior to LBP computation to enhance the gradient information. Local edge patterns (LEP) [103] is an example of this type, in this method the local edge patterns (LEP) are used with color features for color texture retrieval. In LEP, the Sobel edge detection and thresholding are used to find strong edges, and then LBP-like computation is used to derive the LEP patterns.

Also other types of preprocessing have been applied prior to LBP feature extraction. For example, Patterns of Oriented Edge Magnitudes (POEM) [104] 1)firstly computes the gradient of the input image, 2) then incorporates gradient information from neighboring pixels by computing a local histogram of gradient orientations over all cell pixels. 3) the gradient magnitude at the pixel is computed. 4) finally, encodes the accumulated magnitudes using LBP operator within a block. The advantage of this method is computationally efficient.

3.3.2 Neighborhood topology and sampling

One important factor which makes the LBP approach so flexible to different types of problems is that the topology of the neighborhood from which the LBP features are computed can be different, depending on the needs of the given application.

The traditional LBP method identifies a neighborhood as a set of pixels on a circular ring. However, there have been many different neighborhood topologies defined, Figure 3.7 illustrates such a variety.

The extraction of LBP features is usually done in a circular or square neighborhood. A circular neighborhood is important especially for rotation-invariant operators. However, in some applications, such as face recognition, rotation invariance is not required, but anisotropic information may be important. To exploit this, Liao *et al.* proposed Elongated Local Binary Patterns (ELBP) [105] for face recognition. For this specific application domain, the anisotropic information is more important than rotation invariance. Comparing with the conventional LBP, anisotropic structures of the facial images can be captured effectively by the proposed approach using elongated neighborhood, an ellipse shape (Figure 3.7 (c2)). There are three parameters related to the ELBP approach: (1) The long axis of the ellipse; (2) The short axis of the ellipse; (3) The number of neighboring pixels. The X and Y coordinates, g_{ix} and g_{iy}, of each neighbor pixel $g_i \quad (i = 1, 2..., m)$ with respect to the center pixel is defined by Equations 3.8 and 3.9 respectively,

$$R_i = \sqrt{\frac{A^2B^2}{A^2sin^2\theta_i + B^2cos^2\theta_i}} \tag{3.8}$$

$$g_{ix} = R_icos\theta_i, g_{iy} = R_isin\theta_i \tag{3.9}$$

where $\theta_i = (\frac{360}{m}(i-1))$. If the coordinates of the neighboring pixels do not fall exactly at the image grid, then the bilinear interpolation technique is applied.

In the literature, a so called Average Maximum Distance Gradient Magnitude (AMDGM) was proposed to effectively capture gradient magnitude information. AMDGM embeds the gray level difference information between the reference pixel and neighboring pixels in each ELBP pattern. It is found that the ELBP and AMDGM features are well complement with each other. To define AMDGM, the concept of distance gradient magnitude (DGM) was firstly introduced. For each ELBP pattern, there are three parameters: A, B and m denoting the long axis, short axis and the number of neighboring pixels. Then, the distance gradient magnitude for each neighboring pixel g_i, given the center pixel g_c, is defined by Equation 3.10,

$$|\nabla_d I(g_i, g_c)| = \frac{|I_{g_i} - I_{g_c}|}{|v_i - v_c|^2} \tag{3.10}$$

where $v = (x, y)$ denotes the pixel position, I_{g_i} and I_{g_c} are the intensities of the neighbor pixel and the reference pixel respectively. Based on the definition of DGM, the maximum distance gradient magnitude $G(v)$ is defined by Equation 3.11,

$$G(v) = max_{g_i} \left|\nabla_d I(g_i, g_c)\right|, i = 1, 2, ..., m \tag{3.11}$$

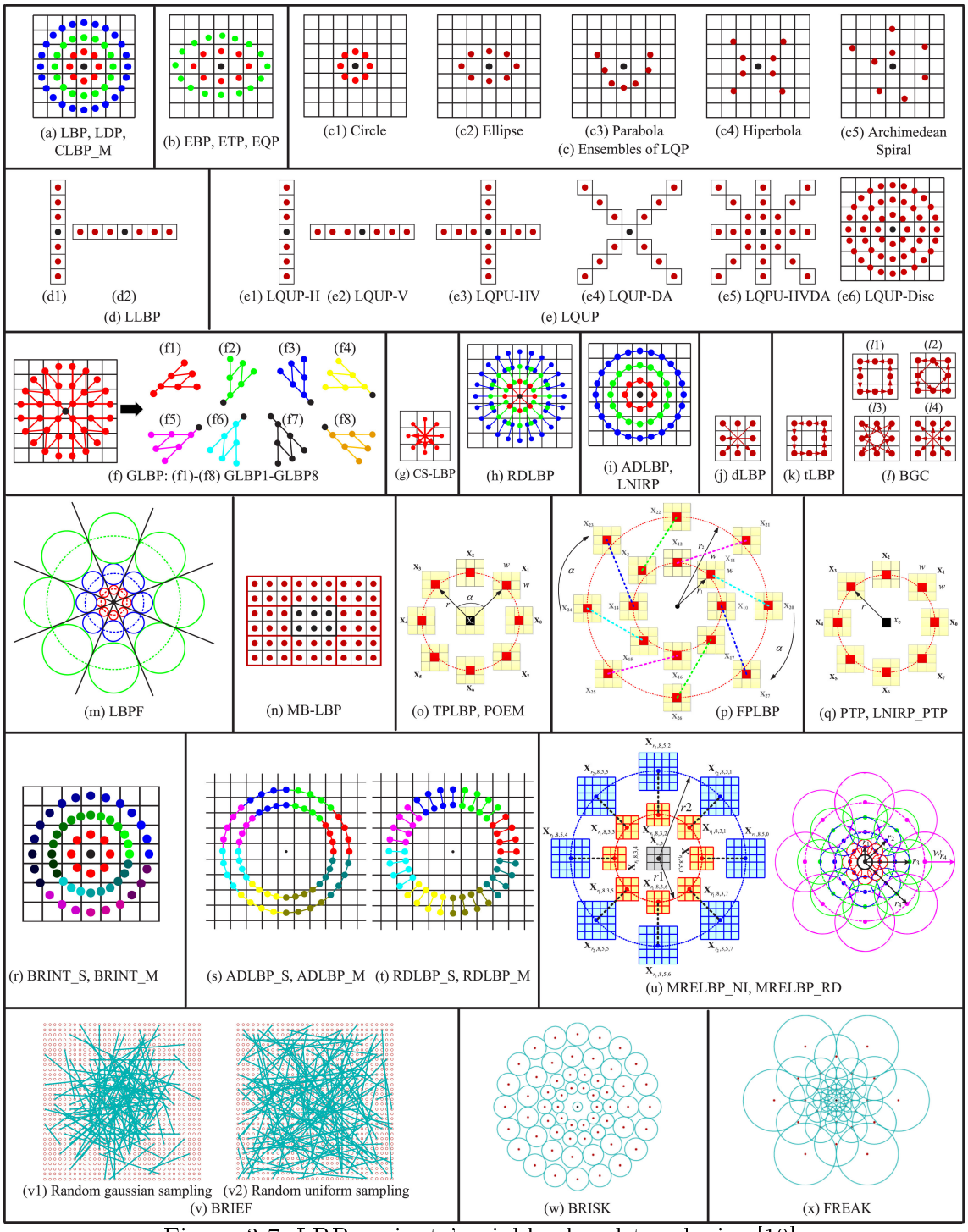


Figure 3.7: LBP variants' neighborhood topologies [10].

Suppose that, in an input image, for each type of uniform ELBP patterns $Pi(i = 1, 2, ..., m)$, its occurrence is N_i. Then, the average maximum distance gradient magnitude (AMDGM) $A(P_i)$ for each type of uniform patterns is defined by Equation 3.12,

$$A(P_i) = \sum_{k=1}^{N_i} \frac{G(v_k)}{N_i} \tag{3.12}$$

where $v_k \in Pi$, the AMDGM feature has more advantage over the conventional gradient magnitude as it encodes the spatial information (i.e. the distance from the neighbor pixel to the reference center pixel) into consideration. It is essential because the neighborhood distribution is no longer isotropic. The distance from each neighborhood pixel to the reference pixel can be different, unlike the conventional LBP. The AMDGM feature is well complement with ELBP because the ELBP provides pattern type distribution and the AMDGM feature implies the general gradient information with spatial information for each type of uniform patterns.

Also, Nanni *et al.* generalized in [106] to parabolic, hyperbolic, and spiral neighborhood topologies, with points detailed in Table 3.1 and in Figure 3.7 (c1-c5) for the neighborhood definition.

Table 3.1: Loci of points used for neighborhood topologies.

Loci of points	Equation	Parameters
Circle	$x^2 + y^2 = r^2$	r is the circle radius
Ellipse	$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$	a, b are the semimajor and semiminor axis lengths
Parabola	$y = -\frac{1}{c}x^2 + 2c$	c is the distance between vertex and focus
Archimedean spiral	$r = a + b\theta$	(r, θ) represent the polar coordinates; the parameter a turns the spiral, while b controls the distance between successive turnings

Another different topology was introduced in the Local Line Binary Pattern (LLBP) [107], which uses vertical and horizontal lines for LBP neighborhood sampling. The basic idea of LLBP is to first obtain the Line binary code along with horizontal and vertical direction separately and its magnitude, which characterizes the change in image intensity such as edges and corners, is then computed. Mathematically,

$$LLBP_h(N, c) = \sum_{n=1}^{c-1} s(h_n - h_c).2^{(c-n-1)} + \sum_{n=c+1}^{N} s(h_n - h_c).2^{(n-c-1)} \tag{3.13}$$

$$LLBP_v(N, c) = \sum_{n=1}^{c-1} s(v_n - v_c).2^{(c-n-1)} + \sum_{n=c+1}^{N} s(v_n - v_c).2^{(n-c-1)} \tag{3.14}$$

$$LLBP_m = \sqrt{LLBP_h^2 + LLBP_v^2} \tag{3.15}$$

where $LLBP_h, LLBP_v, LLBP_m$ are $LLBP$ on horizontal direction, vertical direction, and its magnitude respectively. N is the length of the line in pixel, $c = \lceil \frac{N}{2} \rceil$ is the position of the center pixel h_c on the horizontal line and v_c on the vertical line, h_n is the pixel along with the horizontal line and v_n is the pixel along with the vertical line, and $s()$ is the LBP sign function ($s(x) = 1 \quad if \quad x \geq 0, \quad otherwise \quad s(x) = 0$).

Further generalized in the Local Quantized Pattern (LQP) [108], which exploits vector quantization and lookup tables to allow a larger or deeper neighborhood and more quantization levels. By sampling larger local neighborhoods (Figure 3.7, (d)-(e)), LQP is expected to provide an increase in discriminative power. However, the number of different LBP patterns increases exponentially with the size of the spatial support of the pattern and the number of quantization levels.

Many different neighborhood geometries were used by LQP, these are horizontal (H), vertical (V), diagonal (D) and antidiagonal (A) strips of pixels; combinations of these like horizontal-vertical (HV), diagonal-antidiagonal (DA) and horizontal-vertical-diagonal-antidiagonal (HVDA); and traditional circular and disk-shaped regions (see Figure 3.7, (d)-(e) for sampling topologies). By default LQP compares each noncentral pixel to the central one, but LQP also tests "center symmetric" (CS) codes, where each pixel is compared to the diametrically opposite one. Geometries will be described by notation such as HV_7^3, where HV denotes the neighborhood shape (here a horizontal-vertical cross), the subscript indicates the neighborhood diameter (here 7 pixels) and the superscript indicates the quantization level (here native ternary coding - 3* denotes split ternary coding and 2 binary coding).

3.3.3 Complementary descriptors

The standard LBP encoding rule thresholds all neighboring pixel values against the single, central pixel. This thresholding offers an encoding of a neighborhood with regard to its central value, but all relations between pixels in the neighborhood are lost. Therefore a class of methods has been proposed to generate LBP codes based on simple local differences or magnitudes among neighbors.

Most fundamental was the idea of A Completed Modeling of Local Binary Pattern Operator (CLBP)[24] to decompose local differences into two complementary components: the signs and the magnitudes, corresponding to operators CLBP_S and CLBP_M, respectively. CLBP_S is therefore the same as the original LBP, and CLBP_M offers local contrast (variance) information. Local difference information was proposed in [74], measuring radius differences (RDLBP) and angular differences (ADLBP), as illustrated in Figure 3.7(h, i). Related to ADLBP in taking angular differences are the center-symmetric LBP (CSLBP) [109] of Figure 3.7(g), the binary gradient contours (BGC) [110] of Figure 3.7(l), Transition Local Binary Patterns (tLBP) [111] of Figure 3.7(k), and the directional LBP (dLBP) of Figure 3.7(j). The Local Neighboring Intensity Relationship Pattern (LNIRP) [112] is similar, but is based on second-order derivatives in the circular direction. Although the literature gives the impression that the angular-LBP family has been studied more intensively than radial differences, from [74] the RDLBP is more ef-

fective than ADLBP for texture classification. Rather than dense local neighborhoods, the Geometric Local Binary Pattern (GLBP) explores intensity changes on oriented neighborhoods, meaning a geometry of points on circles with different radii; the neighborhood topology of multi-scale RDLBP and multi-orientation GLBP are nearly the same, the difference lies in the subset of radial differences they choose to generate an LBP code. The Local Derivative Pattern (LDP) is a general framework to encode directional pattern features based on local derivative variations, such that the (n)th-order LDP compares the $(n-1)$th-order directional derivative at the center pixel with those at neighboring pixels. LBP is essentially the first-order special case of LDP. The third-order LBP was found to be most effective in the context of recognition experiments.

3.3.4 Patch-based LBP methods

The traditional LBP methods and many variants are criticized for encoding only local micro-texture and losing nonlocal macro-texture that may be dominant in some cases. By contrast, patch-based LBP variants integrate over larger areas to capture macro-structure information. There are many mechanisms for introducing non-locality, including Three and Four Patch LBP (TPLBP) [113], and Patterns of Oriented Edge Magnitudes (POEM) [114], Binary Rotation Invariant and Noise Tolerant (BRINT) [33], and Median Robust Extended LBP (MRELBP) [34]. These methods are all related, with variations on the shapes of the patches (rectangular, square or pixel arc), filtering (raw pixels or filtered values), the nature of the central comparison (single pixel, patch mean, or patch median), whether one or multiple rings of patches are used, and whether directional or gradient information is captured. LBP has also led to local binary feature descriptors designed for image matching, including Binary Robust Independent Elementary Features (BRIEF) [115], Binary Robust Invariant Scalable Keypoints (BRISK) [116] and Fast Retina Keypoint (FREAK) [117], shown in Figure 3.7(v), (w) and (x) respectively. BRIEF uses random pairwise gray level pixel or Gaussian smoothed pixel comparisons to produce 128, 256 or 512 bit binary patch descriptors for key-point matching. These binary descriptors provide a comparable matching performance with the widely-used region descriptors such as SIFT [11] and SURF [118], but have very short extraction times and very low memory requirements, especially suitable for emerging applications using mobile devices with limited computational capabilities. Comparative evaluations of these descriptors can be found in [119].

3.3.5 Thresholding and quantization

A shortcoming of the original LBP operator is that the thresholding operation directly compares pixel values, making it highly sensitive to noise. Researchers have proposed many LBP variants by changing thresholding schemes to gain noise robustness and discrimination power.

Instead of using only the gray value of the center pixel for thresholding, many other techniques also have been considered, including local means (NILBP [74]), and

one or more shifted thresholds (TMLBP [120]).

Of these methods, ILBP and MBP also code the value of the center pixel, resulting in a doubling in the number of LBP bins. Similarly, the UTGLBP applies a union of multiple TMLBP features computed with different thresholds, thus able to capture a distribution of behavior, but at a cost of a significant increase in feature dimensionality.

The variants can also come form the changes in number of quantization levels. The pioneering approach here is the Texture Spectrum (TS) [92], which actually predates LBP. TS uses an additional parameter, τ, which defines a tolerance for similarity between different gray intensities, allowing for robustness to noise, however at $3^8 = 6561$ bins, TS has a far higher dimensionality than $2^8 = 256$ of those from LBP.

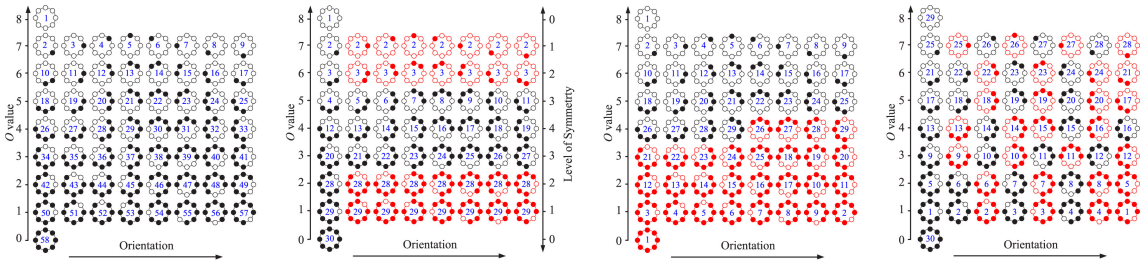
Motivated by TS and LBP, [121] introduced a split coding scheme in order to reduce the dimensionality and proposed LTP. The pair of thresholds leads to an important strength: LTP is capable of encoding pixel similarity modulo noise using the simple rule that any two pixels within some range of intensity are considered similar. Subsequent to [121] many LTP variants were proposed in the literature. In [106] a quinary code was proposed, with two thresholds further split into four binary codes. In [122] the SILBP was proposed to deal with gray scale intensity changes in complex backgrounds, but again at high feature dimensionality. In a related strategy, some researchers proposed to encode small pixel difference as an uncertain bit first and then to determine its value based on the other bits of the LBP code, such as the SoftLBP [123], and Noise Resistant LBP (NRLBP) [124].

FLBP and NRLBP allow multiple LBP patterns to be generated at one pixel position, but at considerably increased computational cost. As a result, the NRLBP [124] was proposed as a lower-complexity improvement on FLBP. In NRLBP, only the uniform patterns are with a simplified contribution weight, which allows the complex method of FLBP to be replaced with a lookup table. Extended LBP (ELBP) presented in [125] generates multiple LBP codes at a pixel position in a different way. The ELBP operator not only encodes the sign information between the central pixel and its neighbors, but also encodes the magnitudes using some additional binary units in an interesting way. The Improved Local Ternary Pattern (ILTP) [126] combines ILBP and LTP in that the LTP code is split into positive and negative ILBP codes.

3.3.6 Encoding and regrouping

The original LBP operator produces rather long histograms of 2^p distinct patterns, overwhelmingly large even for small neighborhoods, leading to poor discriminative power and large storage requirements. It is clear that not all local patterns are meaningful for modeling the characteristics of textures or other image types, thus researchers have attempted to obtain more discriminative, robust and compact features by identifying the most informative pattern groups according some criteria: (1) Heuristic groupings, (2) Co-occurrence groupings, (3) Learning strategies.

Encoding can be based on heuristically grouping LBPs into classes. The goal here



(a) Uniform LBPs (b) SULBP (c) CULBP (d) sLBP

Figure 3.8: Examples illustrating four closely-related LBP variants. Empty and filled circles correspond to bit values of 0 and 1, respectively. The number in the center of each pattern is just a category label, with red patterns indicating the collecting of multiple patterns into one group. Thus the uniform LBP has 58 patterns (a), 30 groups in SULBP (b), 29 different groups in CRLBP (c), and 30 different groups in semantic LBP (d). (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.) [10].

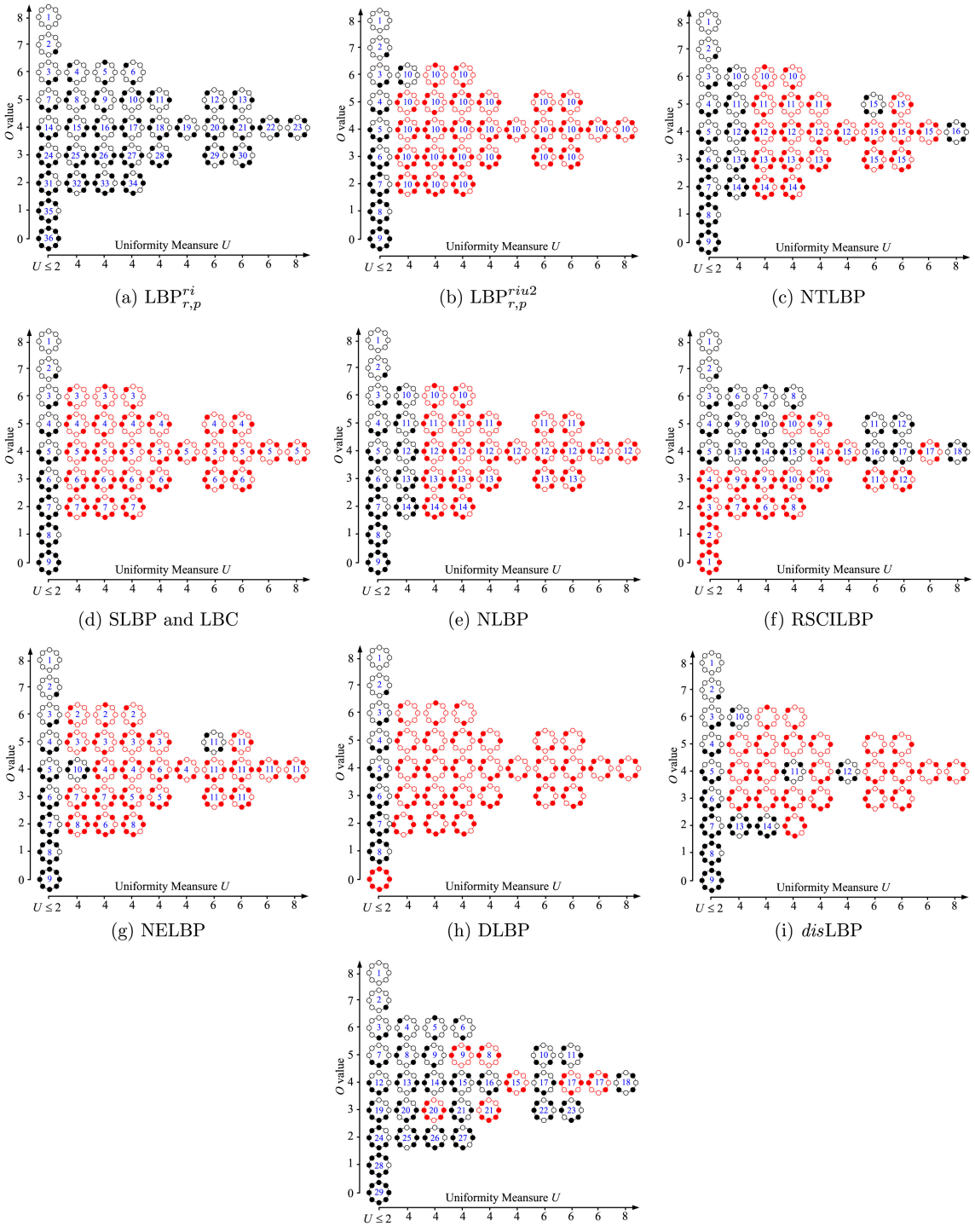
is to attempt to develop a different grouping scheme to regroup all LBP patterns to improve on the traditional descriptors. Figure 3.8 and Figure 3.9 visually depict the grouping behavior for a number of methods.

Symmetric Uniform Local Binary Pattern (SULBP) [127], Non Redundant LBP (NRLBP) [128] and semantic Local Binary Pattern (sLBP) [129], all illustrated in Figure 3.8, share the common idea in regrouping the uniform LBPs to obtain lower feature dimensionality and higher discriminative ability than $LBP^{u2}_{r,p}$. Thus SULBP asserts that uniform patterns representing "edg" and "corne" occur more frequently in face images than those representing "line en" and are more discriminative, NRLBP [130] is designed to be robust to inverted changes of the background and foreground intensities for object detection, and sLBP [129] groups the uniform patterns with similar arch length (number of consecutive 1s) and close orientations together.

Noise Tolerant Local Binary Pattern (NTLBP) [131], Sorted Local Binary Pattern (SLBP) [132], Local Binary Count (LBC) [133], and sorted consecutive Local Binary Pattern_Sign (scLBP_S) [134] are closely related LBP variants which are designed to make better use of the nonuniform patterns instead of discarding them, with illustrations given in Figure 3.9. The groups in scLBP_S are highly similar to those in $LBP^{ri}_{r,p}$ [134]. Similar to scLBP_S, scLBP_M, which is derived from CLBP_M [24], was also proposed in [134].

Another way of LBP encoding is the aggregation of LBP codes depending upon Co-occurrence. The use of co-occurrences of LBP (CoLBP) patterns is borrowed from Gray Level Co-occurrence Matrices (GLCM) [26]. The general idea is to consider the joint probability of pairs of LBPs at certain relative displacements.

Traditional methods for making the GLCM matrix rotation invariant can be applied to CoLBP, such as summing multiple CoLBP over orientation. However, there is an essential difference between applying GLCM to image gray values and to LBP codes; in particular, unlike a gray value, an LBP pattern has spatial structure and orientation. Researchers have thus proposed methods to achieve rotation invariance either at the local feature level or at the co-occurrence level, or both. The key for



(a) $\text{LBP}^{ri}_{r,p}$ (b) $\text{LBP}^{riu2}_{r,p}$ (c) NTLBP

(d) SLBP and LBC (e) NLBP (f) RSCILBP

(g) NELBP (h) DLBP (i) *dis*LBP

(j) scLBP

Figure 3.9: A continuation from Fig. 3, with shading indicating bit value and numbers indicating category labels, comparing nine strategies in space of uniformity $U(4)$ and O value denotes the number of 1s in a pattern. Empty and filled circles correspond to bit values of 0 and 1, respectively. For example, the non-uniform LBP codes, with a uniformity $U > 2$, may be preserved (a), lumped into a single category (b), divided into a few categories (e), or not be preserved at all (h) [10].

the computation of rotation invariant at the co-occurrence level is to construct a rotation invariant coordinate system for each pair of LBP patterns.

Pairwise Rotation Invariant Co-occurrence LBP (PRICoLBP) [135] was proposed to achieve global rotation invariance by sampling the neighboring point along the unit gradient direction or the unit normal direction at the center point. The method was further extended PRICoLBPg to multi-scale and multi-orientation analysis by sampling multiple neighboring points along the gradient direction and norm direction at the center pixel, but resulting in a fairly high dimension feature. Instead of considering co-occurrences of LBPs at different locations, some strategies encode the pairwise information of LBPs at the same location but from various scales, such as Multiscale Joint encoding LBP (MSJLBP) [136].

LBP method can also be encoded by learning their discriminative labels. This encoding learns the most reliable and discriminative dominant patterns; representative work along this line includes Dominant Local Binary Pattern (DLBP) [71], Discriminative Local Binary Pattern (disLBP) [76], Labeled Dominant RICoLBP (LDRICoLBP) [137] and Learning discriminative LBP patterns with AdaBoost (ABLBP) [138].

DLBP [71] uses the most frequently occurring LBP patterns while discards those rarely occurring ones by examining the occurrence frequencies of the rotation invariant LBP groups. disLBP [76] improves on DLBP by considering the intra-class similarity and inter-class distance during learning. Similarly, LDRICoLBP [137] learns co-occurrencing LBP patterns by using DLBP and disLBP's learning schemes. ABLBP [138] uses AdaBoost to select the most discriminative LBP pattern groups.

Almost all heuristic regrouping methods have fairly low feature dimensionality, which is beneficial when generalizing to multiscale analysis and does not increase the computational complexity over traditional LBP methods. In contrast, the co-occurrence LBP methods attempt to increase discriminative power by including spatial co-occurrence information with a considerable dimensionality increase, resulting in limiting the number of neighbors to be at most eight for multiscale analysis. While the learning based methods in have moderate feature dimensionality, they require a pre-training step.

3.3.7 Combining with complementary features

A common branch in the development of new effective local image and video descriptors is to combine the strengths of complementary descriptors. From the beginning the LBP operator was designed as a complementary measure of local image contrast. In many recent studies proposing new texture descriptors the role of the LBP contrast has not been considered when comparing LBP to the new descriptor. The use of LBP (or its simple robust version using a non-zero threshold [120], can still be the method of choice for many applications, and should be considered when selecting a texture operator to be used. An interesting alternative for putting the local contrast into the one-dimensional LBP histogram was proposed by Guo *et al.* [72].

One category of complementary descriptors is to combine with other preprocessing techniques. A high performance preprocessing technique proposed by Vu *et*

al. [102], the biologically-inspired filtering (BF) approach, belongs to this category of complementary descriptors. This technique simulates the performance of human retina in such a way that it applies a DoG filter to detect the "edges", and then splits the filtered image into two "maps" alongside edges. The feature extraction step is then carried out on the two "maps" instead of the input image. Another widely used preprocessing method before LBP computation is Gabor filtering. This is because Gabor filtering and LBP provide complementary information: LBP captures small and fine details, while Gabor filters encode appearance information over a broader range of scales. The pioneering work is LGBP [101]. It filters an image with multi-scale and multi-orientation Gabor filters and then extracts LBP features from Gabor magnitude images. A downside of LGBP includes its high dimensionality and the extra computation burden introduced by filtering. Following LGBP, Gabor preprocessing has also been coupled with other LBP variants to further boost recognition performance, such as Local Quantized Patterns (LQP) [108]. Zhang *et al.* [139] proposed Local Energy Pattern (LEP) for texture classification, where multi-scale and multi-orientation Gaussian-like second-order derivative filters are used to filter the original image. Different from LGBP where each filter response map is converted to one LBP feature map, the filter responses from the same scale but across all orientations are used to generate a LBP type feature map. LEP encodes the relationship among different feature channels using N-nary coding scheme instead of binary coding scheme. A downside of LEP is that it requires pretraining. Similar to LEP, BSIF [140] describes each pixel's neighborhood by a binary code which is obtained by first convolving the image with a set of linear filters and then binarizing the filter responses. The bits in the code string correspond to binarized responses of different filters. However, BSIF learns the filters by utilizing statistics of natural images instead of a manually predefined set of filters. Qian *et al.* [73] proposed first to transform an image into spatial pyramid domain and then to compute the traditional LBP descriptors, and named their method as Pyramid LBP (PLBP).

Li *et al.* [141] proposed an LBP variant named Scale and Rotation Invariant SubUniform LBP ($LBP_{r,p}^{sri\_su2}$) for texture classification. The radius for computation of an local binary pattern at each pixel is defined as the characteristic scale of the local patch centered at that pixel. Different pixels have different characteristic scales, resulting in local scale invariance. The subuniform pattern groups defined by Li *et al.* [141] are essentially the same as the rotation variant uniform groups in $LBP_{r,p}^{u2}$. After obtaining the subuniform histograms, a circular shift LBP histogram is computed to obtain rotation invariance. Note that the circular shifting of the subuniform histograms is actually the same as the glob al matching scheme proposed in [89], which was applied in $LBPV_{r,p}^{u2}GM_{PD2}$. Li *et al.* [141] showed a good discrimination capability of $LBP_{r,p}^{sri\_su2}$ for texture classification. Davarzani *et al.* [142] proposed Weighted Rotation and Scale Invariant LBP (WRSI LBP) to address rotation and scale variations in texture classification. To achieve rotation invariance, dominant orientation needs to be estimated for each pixel in a image. To achieve scale invariance, an approach similar to the one used in $LBP_{r,p}^{sri\_su2}$ [141] is used. Furthermore, Davarzani *et al.* [142] used the minimum magnitude of local differences as an adaptive weight to adjust the contribution of an LBP code in histogram

calculation, resulting WLBP operator. One downside of $LBP_{r,p}^{sri\_su2}$ and WRSI LBP is that characteristic scale and dominant orientation estimation is computationally expensive and unreliable.

Complementary descriptors can also be formed by combining multiple LBP-like codes. Guo *et al.* [24] can be known as the first authors who combine multiple LBP type features. They proposed to decompose the image local differences between a center pixel and its neighbors into two complementary components, the signs (s_p) and the magnitudes (m_p), and therefore two operators CLBP_S and CLBP_M were proposed.

$$s_p = s(g_p - g_c),\ m_p = |g_p - g_c| \tag{3.16}$$

where g_p, g_c, and $S(x) = \begin{cases} 1 & x \geqslant 0 \\ 0 & x < 0 \end{cases}$.

Two operators called CLBP-Sign ($CLBP\_S$) and CLBP-Magnitude ($CLBP\_M$) are encoded, where the $CLBP\_S$ is equivalent to the conventional LBP, and the $CLBP\_M$ measures the local variance of magnitude. The $CLBP\_M$ is defined as follows:

$$CLBP\_M_{P,R} = \sum_{p=1}^{P-1} t(m_p, c)2^p,\ t(x,c) = \begin{cases} 1, & x \geq c \\ 0, & x < c \end{cases} \tag{3.17}$$

where the threshold c is set as the mean value of m_p of the whole image. Guo *et al.* observed that the center pixel, which expresses the image local gray level, also has discriminative information. Thus, they defined an operator named CLBP-Center ($CLBP\_C$) to extract the local central information as follows:

$$CLBP\_C_{P,R} = t(g_c, c_I) \tag{3.18}$$

where threshold c_I is set as the average gray level of the whole image. By combining the three operators $CLBP\_S$, $CLBP\_M$ and $CLBP\_C$, significant improvement is made for texture classification compared to the original LBP.

The Completed Local Binary Count (CLBC) approach [133] is highly similar to CLBP, but with a different binary patterns regrouping scheme. Guo *et al.* [76] combined CLBP and disLBP to introduce disCLBP. Liu *et al.* introduced CINIRD in [74], which is the joint distribution of NILBP, RDLBP and CILBP, turned out to be very discriminative.

CLBP, CLBC and CINIRD are highly sensitive to image noise, high in dimensionality and being too local. To overcome these shortcomings, Liu *et al.* recently proposed Binary Rotation Invariant and Noise Tolerant (BRINT) [143] and Median Robust Extended LBP (MRELBP) [34]. BRINT combines three individual descriptors BRINT_S, BRINT_M and BRINT_C. Unlike CLBP and CINIRD where only rotation invariant uniform patterns are considered, BRINT uses all the rotation invariant patterns to avoid the risk of the uniform patterns not taking the dominant proportion. In BRINT, pixels are sampled in a circular neighborhood, but keeping the number of bins in a single-scale LBP histogram constant and small, such that

arbitrarily large circular neighborhoods can be sampled and compactly encoded over a number of scales. BRINT can extract features from a number of scales with low feature dimensionality. Moreover, BRINT was showed to be robust to noise.

In order to encapsulate microtexture and macrotexture information at the same time and further to enhance the discriminative power and noise robustness, Liu *et al.* [34] presented MRELBP to combine MRELBP_NI, MRELBP_RD and MRELBP_CI. Although MRELBP was derived based on CINIRD, it outperformed CINIRD significantly, especially in random noise corrupted situations. Moreover, MRELBP has much lower feature dimensionality than other methods in this subcategory. Ryu *et al.* [134] developed a method to represent a local binary pattern as a feature vector in Euclidean space by sorting numbers of consecutive patterns. In [134], they applied the idea to CLBP_S and CLBP_M, resulting in three new LBP type features scLBP_S, scLBP_M+ and scLBP_M- (CLBP_M was separated into two features) which was concatenated into a single feature vector named scLBP. Subsequently, Ryu [134] introduced dictionary learning of scLBP based on kd-tree which separates data with a space partitioning strategy. Guo *et al.* [144] proposed Scale Selective LBP (SSLBP) with the aim of addressing scale variation for texture classification. In SSLBP, a scale space of a texture image is firstly derived by a Gaussian filter; Then a histogram of pre-learned dominant binary patterns is built for each image in the scale space; Finally, for each pattern, the maximal frequency among different scales is considered as the scale invariant feature. Essentially, SSLBP combines the ideas of DLBP [71] and CLBP (CLBP_CS and CLBP_CM) [24] to learn the dominant binary patterns.

Finally, combining LBP and other complementary features is also a popular approach. From the very beginning the original LBP operator was designed as a complementary measure of local image contrast, and the joint histogram of the two complementary features, namely LBP/VAR, was proposed for rotation invariant texture classification. However, the value of VAR is continuous so that a quantization step is needed to calculate the histogram.

Ahonen *et al.* [144] proposed LBP Histogram Fourier features (LBPHF) to achieve rotation invariance globally by combining LBP and Discrete Fourier transform (DFT). LBPHF is derived by first computing a uniform LBP histogram over the whole image, and then constructing rotationally invariant features from the DFT transform of the histogram. Later in [75], LBPHF is combined with the CLBP_S and CLBP_M descriptors [24] to further improve its distinctiveness.

Guo *et al.* [72] proposed LBPV to incorporate the local contrast information into the LBP histogram by utilizing the variance VAR as a locally adaptive weight to adjust the contribution of each LBP code. LBPV avoids the pretraining step for VAR quantization used in [69]. Likewise, Guo *et al.* [72] also obtained rotation invariance globally. They estimate the principal orientation of an image from the uniform LBP histogram and then use the estimated principle orientation to align the LBPV features. This raises two issues. For one, it is unstable and inaccurate to estimate the dominant orientation of a texture image since lots of texture images do not have an obvious principle orientation. For another, their proposed global matching procedure have to handle high dimensional feature. Arguing that local variance feature VAR is

an isotropic measurement and fails to capture orientation information in textures, Guo *et al.* developed Adaptive Local Binary Pattern (ALBP) by incorporating the directional statistical features (i.e. the mean and standard deviation of the local absolute differences) for rotation invariant texture classification. In addition, the least square estimation is used to adaptively minimize the local difference for more stable directional statistical features.

Similar to LBP variance (LBPV) [72], Discriminative Robust Local Binary Pattern (DRLBP) [145] is proposed to combine LBP and gradient magnitude information. Instead of considering the LBP code frequencies, the pixel gradient magnitude is used as a weight to be assigned to each code which is then voted into the bin that represents the code. DRLBP is formulated by concatenating two histogram vectors: the difference histogram which is the absolute difference between the bins of an LBP code and its complement and the sum histogram which is the sum of the bins of an LBP code and its complement. It is hoped that DRLBP can solve the problem of discrimination between a bright object against a dark background and vice versa inherent in LBP. Satpathy et al. [145] also extend the idea of DRLBP to LTP and proposed DRLTP approach.

In addition to applying LBP to Gabor filtered face images [101], the fuse of LBP variants and Gabor features has also been explored, with applications in texture classification [71] and face recognition [70]. Wang et al. [146] combined Histogram of Gradients (HOG) with LBP, performing very well in human detection with partial occlusion handling. In addition to HOG and LBP, Hussain and Triggs [147] used LTP. Klare and Jain [148] exploited the combination of LBP and SIFT for heterogeneous face recognition. Chan *et al.* [149] fused LBP and Local Phase Quantization (LPQ) for face recognition with robustness to image degradation caused by illumination variations and blurring.

3.3.8 Other methods inspired by LBP

LBP has inspired the development of related local image descriptors, including Local Phase Quantization (LPQ) [150], Weber Law Descriptor (WLD) [151], Local Higher-Order Statistics (LHS) [152], Local Frequency Descriptor (LFD) [153] and Discriminant Face Descriptor (DFD) [154].

LPQ [150] is generated by quantizing the Fourier transform phase in local neighborhoods, such that histograms of LPQ labels computed within local regions are used as a texture descriptor similar to LBP. LPQ is, by design, tolerant to most common types of image blurs. WLD [151] is designed 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. WLD [151] is a 2D histogram of differential excitation and gradient orientation, where the differential excitation reflects the ratio between the relative intensity differences of a center pixel against its neighbors and the intensity of the center pixel itself. WLD demonstrated good results on texture classification and face detection.

Lategahn *et al.* [155] applied a filter bank to transform a texture image into multidimensional filter response space and subsequently estimated the joint probability

density functions of the filter responses by Gaussian mixture models (GMM). Two types of filter banks, i.e. the oriented difference filters of the LBP method [156] and wavelet frame transform filter bank, were considered. This method avoids the crude quantization errors of LBP. Motivated by the idea of Fisher Vector [157], Sharma *et al.* [152] proposed Local Higher-Order Statistics (LHS) of oriented difference filter responses of the LBP method [156] for image description. Like the method proposed by [155], the LHS requires neither any user specified quantization of the feature space nor any heuristics for discarding low occupancy volumes of the space. Experiments with texture and face databases demonstrate good performance.

Motivated by LBP, Maani *et al.* [153] proposed Local Frequency Descriptor (LFD) for texture classification. The LFD descriptor [153] is based on the local frequency components that are computed by applying 1D Fourier transform on the neighboring pixels on a circle centered at each pixel. The low frequency components are kept since they are the major constituents of the circular neighbors and can effectively represent textures. Then three sets of features are extracted from the low frequency components, two based on the phase and one based on the magnitude. LFD is invariant to rotation and linear changes of illumination. Maani *et al.* [158] also extend the idea of LFD and proposed Robust Edge Aware Descriptor (READ) for image matching. The recent DFD descriptor learns the most discriminating local features, computing the difference vectors between the center patch and each of its neighboring patches to form a pixel difference matrix, which is then projected and re-grouped to form the discriminant pattern vector. A standard bag of words model [52, 159] is applied for face representation. The DFD descriptor performs well in face identification and verification, however it involves a number of parameters, has high dimensionality, a time consuming training process, and high training data needs.

3.4 Conclusions

This chapter has reviewed LBP as an image feature method which mainly has two stages. In the first step, it transforms an image into an array or image of integer labels describing small-scale appearance of the image. In the second step, the statistics of the transformed image, most commonly the histogram, are then used as features for further analysis. The relationship between LBP and other close-related methods has been also recalled. Finally, the chapter has covered different versions of the actual LBP operator in spatial domain which is commonly used for texture classification and close-related applications. Next chapter will review an alternative approach for image representation in general and texture features in particular, the Scattering transform method.

Chapter 4

Scattering Transform

The previous chapter reviews a well-know feature method, which captures the small scale appearance of the image, the LBP method. In this chapter, an alternative feature approach which extracts larger scale information of image appearance will be reassessed. We therefore present the scattering transform, its construction, and the scattering network (ScatNet) implementation of scattering transform. ScatNet is a cascade of wavelets transforms followed by a modulus non-linearity, which is spatially averaged. The final averaging permits to build invariance to translation and to build stable invariants to small deformations. More recent, convolutional neural network (CNN) emerges as a frontier in the domain of image classification. In addition, ScatNet and convolutional neural network (CNN) to a certain extend are similar except the fact that ScatNet does not have the learning stage in it. We therefore present at the end of this chapter the main points of a CNN architecture and some of its well-known examples.

4.1 Wavelet transform

A family of wavelets can be obtained from a set of mother wavelets ψ_θ by dilating each mother wavelet with dyadic scales $1 \leq 2^j < 2^J$

$$\psi_{\theta,j}(u) = 2^{-2j}\psi_\theta(2^{-j}u) \tag{4.1}$$

The multiplicative constant 2^{-2j} is chosen so that the dilated wavelet verifies

$$\overline{\psi}_{\theta,j}(\omega) = \overline{\psi}_\theta(2^j\omega) \tag{4.2}$$

where $\overline{\psi}$ if the Fourier transform of ψ. The wavelet coefficients of an image consist in all convolutions $\left\{x \star \psi_{\theta,j}\right\}_{\theta,j}$ of the image with the wavelets $\psi_{\theta,j}$ where the convolution is defined as

$$x \star \psi_{\theta,j} = \int_{v\in\mathbb{R}^2} x(v)\psi_{\theta,j}(u-v)dv. \tag{4.3}$$

The wavelet coefficients $\{x \star \psi_{\theta,j}\}_{\theta,j}$ retain the variations of the signal x at all scales smaller than 2^J. Complementary to these variations, a coarse approximation of the signal is captured through convolution with a single low-pass window ϕ_J, which is obtained by dilating at scale 2^J

$$\phi_J(u) = 2^{-2J}\phi(2^{-J}u) \tag{4.4}$$

where ϕ is typically a Gaussian window $\phi = e^{\frac{-|u|^2}{2\sigma^2}}$. In this case, ϕ_J is a Gaussian window of width $2^J\sigma$.

The wavelet transform $W(x)$ of a signal x consists in both the local average and the wavelet coefficients

$$W(x) = \{x \star \phi_J, x \star \psi_{\theta,j}\}_{\theta,j} \tag{4.5}$$

4.2 Oriented Gabor and Morlet wavelets

Among all possible sets of mother wavelets, the oriented one will be in favor for building rotation invariants. They are constructed as rotated versions of a single mother wavelet ψ.

$$\psi_\theta(u) = \psi(r_{-\theta}u) \tag{4.6}$$

so that the family of multiscale wavelets $\{x \star \psi_{\theta,j}\}_{\theta,j}$ is indexed by (θ, j) its elements are defined as rotated and dilated versions of a single mother wavelet ψ

$$\psi_{\theta,j}(u) = 2^{-2j}\psi(2^{-2j}r_{-\theta}u) \tag{4.7}$$

A two dimensional wavelet is said to be analytic if its Fourier transform has a support which is contained in a half space of $\mathbb{R}^2$. Analytic wavelets have their complex modulus which tends to be much more regular than their real or imaginary part. A typical example of such wavelet is the Gabor wavelet

$$\psi_{Gabor}(u_1, u_2) = \frac{1}{2\pi\sigma^2} e^{\frac{-|u|^2}{2\sigma^2} + i\xi u_1} \tag{4.8}$$

A Gabor wavelet consists in a Gaussian envelope of width σ modulated by an horizontal complex sine wave of frequency ξ. Its Fourier transform is

$$\overline{\psi}_{Gabor}(\omega) = e^{-2\sigma^2|\omega - \xi(0,1)^T|^2}. \tag{4.9}$$

It is a Gaussian of width $(2\sigma)^{-1}$ centered around $\xi(1,0)^T$. Therefore a Gabor wavelet is not exactly analytic because the support of a Gaussian is infinite but if σ/ξ is sufficiently small, then it is a good approximation to say that the Fourier transform $\overline{\psi}_{Gabor} = 0$ for $\omega \leq 0$.

Gabor wavelets are the decent theoretical trade off between spatial and frequency localization. They also have a hermitian symmetry

$$\psi_{Gabor}(-u) = \psi_{Gabor}(u)^* \tag{4.10}$$

where $*$ denotes the complex conjugate. Therefore, when computing convolutions of a real signal x with rotated Gabor wavelets, the wavelet coefficients can be deduced for $\theta \in [\pi, 2\pi)$ from those for which $\theta \in [0, \pi)$ because

$$x \star \psi_\theta(u) = (x \star \psi_{\theta+\pi}(u))^* \tag{4.11}$$

this will save approximately a factor of two of computational time and memory usage. In addition, if we are only interested in the complex modulus, as it will be the case for scattering networks, we can discard all the orientations $\theta \in [\pi, 2\pi)$ because

$$|x \star \psi_\theta(u)| = |x \star \psi_{\theta+\pi}(u)| \tag{4.12}$$

In scattering network [35, 160, 161], a variation of Gabor wavelets is used, the elongated Morlet wavelet. Because Gabor wavelets have a non-zero average which makes their responses non-sparse. Morlet wavelet fixes this while preserving the smoothness and spatial localization of a Gabor wavelet by subtracting a Gaussian envelope multiplied by an appropriate constant K. The Morlet wavelet and is defined as

$$\psi_{Morlet}(u) = \frac{1}{2\pi\sigma^2} e^{\frac{-|u|^2}{2\sigma^2}} (e^{i\xi u_1} - K) \tag{4.13}$$

The constant K is computed such that $\int \psi_{Morlet} = 0$. Because of this subtraction of a low pass envelope, Morlet wavelets are slightly less analytic and less localized in frequency than Gabor wavelets, but they have an exact zero-mean.

Computing the wavelet transform with Morlet wavelets requires to discretized θ in $2L$ of orientations between $[0, 2\pi)$. When choosing a larger L, Sifre *et al.* [161] modified the wavelets to increase the angular sensitivity, otherwise the extra orientations do not bring additional information. For this reason, they replace the circular envelope of the Gaussian envelope with an elliptical one whose horizontal and vertical semi axes are respectively σ and σ/s.

$$\psi_{ElongatedMorlet}(u_1, u_2) = \frac{1}{2\pi\sigma^2/s} e^{-\frac{1}{2\sigma^2}(u_1^2 + u_2^2 s^2)} (e^{i\xi u_1} - K) \tag{4.14}$$

Sifre *et al.* [161] typically choose $s \propto L^{-1}$ where L is the number of orientations so that the horizontal wavelet $\theta = 0$ has a growing vertical semi axes proportional to L and thus becomes more sensitive to the horizontal orientation when the number of orientations L increases. Several scales per octave Q can be computed instead of only one. In this case typically $\sigma/\xi \propto 2^Q$ is chosen to increase the scale sensitivity of wavelets when the number of scales per octave rises. Figure 4.1 compares two families of elongated Morlet wavelets with different number of scales J, orientations L and scales per octave Q, along with their associated Littlewood Paley function. One can see that in all configurations, the Littlewood Paley has reasonable bounds with $\alpha \approx 0.7$. If the number of orientations and scales per octave of wavelets increases it will rise their localization in the Fourier plane. This makes a particular wavelet $\psi_{\theta,j}$ is slightly redundant with its adjacent neighbors $\psi_{\theta\pm\pi/L, j\pm1/Q}$ and almost orthogonal to all other wavelets.

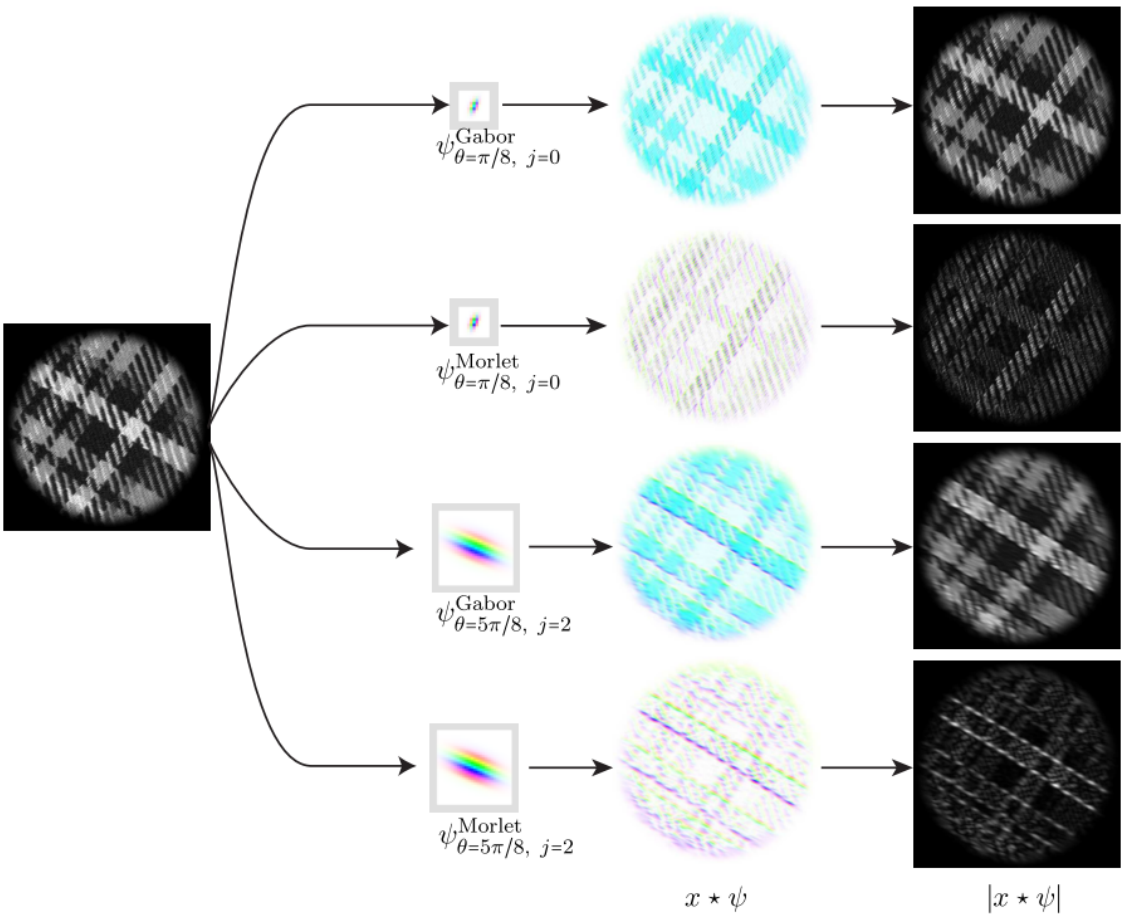


Figure 4.1: Comparison of elongated Gabor and Morlet two dimensional wavelets. The left image is filtered with Gabor and Morlet wavelets obtained with constant window spread $\sigma = 0.8$, a top frequency $\xi = 3\pi/4$ and a slant $s = 0.5$, at two pairs of orientation and scale. The image is filtered and which result in a complex signal $x \star \psi$ of which we compute the modulus $|x \star \psi|$. The Gabor wavelets have a non-zero DC component, which tends to dominate over the high frequencies [161].

When increasing the number of orientations and scales per octave, elongated Morlet wavelets have better angular and scale sensitivity and therefore better separate the different frequency components of an image. This can positively impact classification, but it also has a number of disadvantages. The most obvious one is the rise in number of convolutions to compute a wavelet transform. Since the wavelets are better localized in Fourier space, they are also less well localized spatially. This will grow their spatial support and therefore increase the time required to perform a single convolution, if computed in the spatial domain. As explained in [161], better Fourier localization also means less stability to deformation, which can adversely affect classification at a certain point. Therefore, the choice of the number of orientations and scales per octave is a trade off between the better separability and worsen instability to deformations of the wavelet transform, and the resources we are willing to dedicate to its computation.

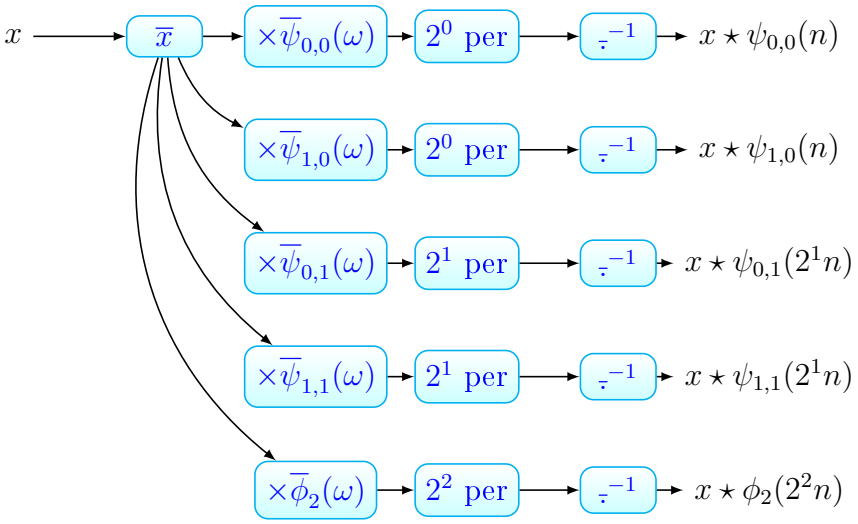


Figure 4.2: Fourier implementation of the wavelet transform with $J = 2$ scales and $L = 2$ orientations. The Fourier transform of x is computed first, its output is then multiplied with the Fourier transform of each function in the wavelet family $\psi_{\theta,j}, \phi_J$. The results are periodized according to the target resolution, finally an inverse Fourier transform is applied.

4.3 Wavelet transform fast implementations

This section discusses the two methods which Mallat *et al.* [160, 161] implemented the wavelet transform in their toolbox, the ScatNet. While the Fourier implementation is more flexible, the filter bank strategy is faster but also more restricting. It supposes that the Fourier transform of the wavelets can be written as an infinite produce of Fourier transform of some dilated filters. Its running time is proportional to the support of the filters.

4.3.1 Fourier implementation

A first possible strategy to implement the wavelet transform is to compute all the convolutions in Fourier domain. Given two discrete finite periodic images x and ψ, then

$$\overline{(x \star \psi)}(\omega) = \overline{x}(\omega)\overline{\psi}(\omega) \tag{4.15}$$

where $\overline{f}$ is the Fourier transform of f. To compute the convolutions with all wavelets $\psi_{\theta,j}$, the Fourier transform of x is computed first, then multiply the result with all the Fourier transforms of $\overline{\psi}_{\theta,j}$. The inverse Fourier transforms of the resulting signals are final outputs. A Fourier transform is implemented with a fast Fourier transform (FFT) algorithm which has a complexity of $\mathcal{O}(n \log n)$ for an image of n pixels. If the wavelets are fixed, as it is the case for our applications, their Fourier transform can be precomputed. Also, $x \star \psi_{\theta,j}$ has a spatial regularity and can therefore be subsampled by 2^j. Instead of computing the inverse Fourier transform and then subsampling, it

is equivalent, and more efficient, to periodize the product of the Fourier transforms and then to compute the inverse Fourier transform at the lowered resolution. The algorithm is illustrated in Figure 4.2.

4.3.2 Filter bank implementation

Every wavelet of its family $\{\psi_{\theta,j}\}$ is actually the dilated versions of another. The filter bank implementation method can leverage this fact while the Fourier one can not. A fast wavelet transform [162] takes advantage of this by computing approximations $A_j x$ of the input signal at different resolutions and computing $x \star \psi_{\theta,j}$ by filtering $A_j x$ at the output resolution 2^j.

This assumes that the Fourier transform of the window ϕ_0 at scale 1 and each mother wavelet ψ_θ can be written as a product of Fourier transform of discrete dilated filters h and g_θ

$$\overline{\phi_0}(\omega) = \prod_{j<0} \overline{h}(2^j\omega) \tag{4.16}$$

$$\overline{\psi}_\theta(\omega) = \overline{g}_\theta \overline{\phi}(\omega) \tag{4.17}$$

Initialize $A_0 = x \star \phi_0$, and denote

$$A_j x(n) = x \star \phi_j(2^j n) \tag{4.18}$$

$$B_{\theta,j} x(n) = x \star \psi_{\theta,j}(2^j n) \tag{4.19}$$

for $n \in \mathbb{Z}^2$. In practice, $A_0 x$ is not actually computed, the input image is used instead. Definitions (4.16), (4.17) leads to

$$A_{j+1} x(n) = \sum_p A_j x(2p) h(n-2p) \tag{4.20}$$

$$B_{\theta,j} x(n) = \sum_p A_j x(p) g(n-p). \tag{4.21}$$

Therefore, the subsampled wavelet transform $\left\{x \star \phi_J(2^J n), x \star \psi_{\theta,j}(2^j n)\right\}_{n,\theta,j}$ is implemented as convolutions followed by downsampling. These convolutions are done with filters h and g_θ whose support do not change with the scale 2^j of the wavelet $\psi_{\theta,j}$. Therefore, these convolutions can be implemented in the spatial domain, in the small filter support regime where they are faster than FFT-based convolutions. Equations (4.20), (4.21) are compactly expressed as

$$A_{j+1} x = (A_j x \star h) \downarrow 2 \tag{4.22}$$

$$B_{\theta,j} x = A_j x \star g_\theta \tag{4.23}$$

Let N be the size of image and P be the size of filters. A convolution at the finest resolution is implemented as a weighted sum over P elements for each position n

$$x \star h(h) = \sum_p x(n-p) h(n) \tag{4.24}$$

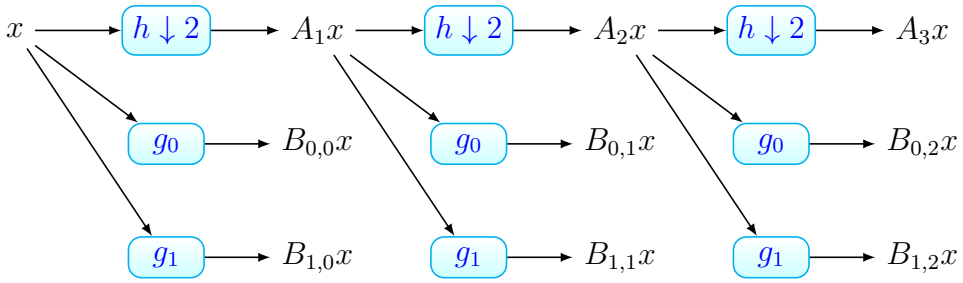


Figure 4.3: Filter bank implementation of the wavelet transform with $J = 3$ scales and $L = 2$ orientations. A cascade of low pass filter h and downsampling results in low frequencies $A_j x = x \star \phi_J$ and filters g_θ output the high frequencies $B_{\theta,j} x = x \star \psi_{\theta,j}$. This cascade leads to a tree whose internal nodes are intermediate computations, and its leaves are the output of the downsampled wavelet transform.

and therefore requires $N \times P$ operations. The cascade requires $1+L$ such convolution at each resolution 2^j. The wavelet transform complexity as shown in [162] is $\mathcal{O}(LNP)$ and requires a memory of $\mathcal{O}(LNP)$ where L is the number of orientations, N is the size of the input image and P is the size of the filters h and g_θ. In practice, filters of size $P = (2q+1) \times (2q+1)$ where $q = 3$ are typically used, Mallat *et al.* shown in [162] that their filter bank implementation is faster their Fourier one. The algorithm is broadly illustrated in Figure 4.3.

4.4 Construction of Scattering transform

4.4.1 Wavelet modulus operator

It is proved in [160] that the averaging filter ϕ_J builds a stability to deformation and translation invariance up to 2^J. However, the averaging only captures the very coarse approximation of x and loses all its high frequency components. A wavelet transform recovers the high frequencies by convolutions with the high pass wavelets $x \star \psi_{\theta,j}$ but these high frequencies are only invariant to translation up to 2^j, not the maximum scale 2^J.

To make sure the resulting wavelet coefficients of $x \star \psi_{\theta,j}$ are invariant to translations up to 2^J while maintaining the stability to deformations, an obvious measure is to average them into $x \star \psi_{\theta,j} \star \phi_J$ but this yields almost zero coefficients because $\psi_{\theta,j}$ and ϕ_J are specifically designed to be almost orthogonal to each other. Therefore, to build non-trivial translation invariance, a possible strategy is to intertwine a non-linearity between the wavelet convolution $\star \psi_{\theta,j}$ and the averaging ϕ_J.

The convolutions of an input signal x with an analytical oriented wavelet such as Gabor or Morlet wavelet results in a complex signal $x \star \psi_{\theta,j}(u)$. The amplitude of this signal is a regular envelope which describes how the signal is correlated to the orientation θ and scale 2^j in the neighborhood of u, while its phase varies almost linearly and is sensitive to local displacements. Figure 4.1 presents the result of such

convolutions. By applying a complex modulus to such filter responses, it will discard local displacements which are irrelevant to describe the image content. It is therefore a good non-linearity operator to apply before averaging, because the averaging will capture more information when its input becomes more regular. Complex modulus also has the advantages of being contractive, preserving the norm and the stability to deformation.

The wavelet modulus operator $W(x)$ consists in the approximation $x \star \phi_J$ coefficients and the modulus of the wavelet coefficients $x \star \psi_{\theta,j}$,

$$|W(x)| = \left\{x \star \phi_J, |x \star \psi_{\theta,j}|\right\}_{\theta,j} \tag{4.25}$$

It decomposes the signal into a first linear invariant $S_0 x$, called scattering of order 0, defined by

$$S_0 x(u) = x \star \phi_J(u) \tag{4.26}$$

which is invariant to translation up to 2^J and stable to deformation, and a first non-linear covariant part

$$U_1 x(u, \theta, j) = |x \star \psi_{\theta,j}(u)| \tag{4.27}$$

which is proved in [160] to be invariant to small translation up to 2^j and stable to deformation.

4.4.2 Scattering operator

A scattering operator computes local image descriptors with a cascade of wavelet modulus operator. The resulting scattering representation is locally invariant to translations. It includes coefficients which are similar to SIFT [11] descriptors, together with co-occurrences coefficients at multiple scales and orientations.

Cascading wavelet transform followed by a non-linearity modulus operator is the strategy used to build homogeneous wavelet up to 2^J, $U_1 x$ is also averaged which yields a second invariant $S_1 x$, called scattering of order 1,

$$S_1 x(u, \theta, j) = |x \star \psi_{\theta,j}| \star \phi_J(u) \tag{4.28}$$

Similar to the case for $S_0 x$ (4.26), the averaging in $S_1 x$ also loses high frequency information of $U_1 x$. This is supplemented later via wavelet modulus coefficients,

$$U_2 x(u, \theta_1, j_1, \theta_2, j_2) = ||x \star \psi_{\theta_1,j_1}| \star \psi_{\theta_2,j_2}(u)| \tag{4.29}$$

Because $U_1 x(., 1, j1)$ is invariant to translations up to 2^{j_1}, its fine scale coefficients $U_1 x(., \theta_1, j_1) \star \psi_{\theta_2,j_2}$ are almost zero for fine scale $j_2 \leq j_1$. Therefore, only coefficients whose scale $j_2 > j_1$ are computed.

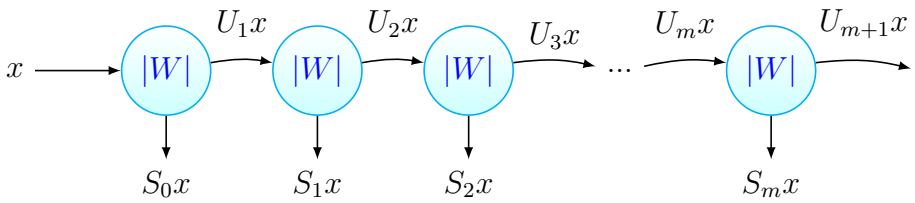


Figure 4.4: Translation scattering can be seen as a neural network which iterates over wavelet modulus operators W. Each layer m outputs averaged invariant S_mx and covariant coefficients $U_{m+1}x$.

U_2x is invariant to translations up to 2^{j_2}, but not to 2^J. Therefore, U_2x is averaged into the second order scattering coefficients

$$S_2x(u, \theta_1, j_1, \theta_2, j_2) = ||x \star \psi_{\theta_1,j_1}| \star \psi_{\theta_2,j_2}| \star \phi_J(u) \tag{4.30}$$

This non-linear decomposition can be generalized to order m as follows

$$\begin{aligned} S_mx(u, \theta_1, j_1, ..., \theta_m, j_m) &= U_m(., \theta_1, j_1, ..., \theta_m, j_m) \star \phi_J(u) \\ &= |...|x \star \psi_{\theta_1,j_1}|... \star \psi_{\theta_m,j_m}| \star \phi_J(u) \\ U_{m+1}x(u, \theta_1, j_1, ..., \theta_{m+1}, j_{m+1}) &= ||U_mx(., \theta_1, j_1, ..., \theta_m, j_m)| \star \psi_{\theta_{m+1},j_{m+1}}(u)| \\ &= |...|x \star \psi_{\theta_1,j_1}|... \star \psi_{\theta_{m+1},j_{m+1}}| \end{aligned} \tag{4.31}$$

Let the scattering path $p_m = (\theta_1, j_1, ..., \theta_m, j_m)$, then 4.31 can be rewritten as follows

$$\begin{aligned} S_mx(u, p_m) &= U_mx(., p_m) \star \phi_J(u) \\ U_{m+1}x(u, p_{m+1}) &= |U_m(., p_m) \star \psi_{\theta_{m+1},j_{m+1}}| \end{aligned} \tag{4.32}$$

then compactly written as

$$\{S_mx, U_{m+1}x\} = |W|U_mx \tag{4.33}$$

The scattering thus successively applies the wavelet modulus operator $|W|$ to decompose the residual layer U_mx into a translation invariant layer S_mx and a non-linear covariant deeper layer $U_{m+1}x$ which will be either re-transformed or simply discarded when it becomes negligible. The scattering structure is illustrated in Figure 4.4.

The scattering invariant vector consists of scattering vector of all orders up to a maximum order M : $Sx = \{S_0x, ..., S_Mx\}$

4.4.3 Scattering network

The scattering vectors are computed by cascading several scattering operators which form the so-called scattering network. The scattering oprators are in turn created

by the wavelet transform whose implementation can be chosen to be either the Fourier or the filter bank of Section 4.3. The first scattering operator computes S_0x, subsampled at resolution 2^J, and $U_1x(.,\theta_1,j_1)$, subsampled at resolution 2^{j_1}. The second scattering operator is applied to every $U_1x(.,\theta_1,j_1)$ at input resolution j_1. This results in L wavelets modulus at each resolution 2^{j_1}, all of which costs $\mathcal{O}(LNP)$ if the filter bank algorithm is used to implement the wavelet transform. Cascading this process results in a time complexity of $\mathcal{O}(LNP)$ and a memory cost of $\mathcal{O}(LNP)$ where L is the number of orientations, M is the scattering order, N is the size of the input and P is the size of the filters. For classification purposes, invariant scattering coefficients Sx are often mostly interested in the while the intermediate computations Ux can be needed for the subsequent layers of the scattering network computation. Because all its components are downsampled by a factor 2^J, storing Sx requires $\mathcal{O}(2-2J(JL)MN$ coefficients, which in practice, is much smaller than the size N of the original image. For the fully translation invariant scattering $2^{2J}=N$ so that the representation has $\mathcal{O}((L/2logN)^M)$ coefficients. Figure 4.4 illustrates scattering network as a special type of convolutioanl neural network.

4.5 Deep convolutional neural networks

A deep convolutional neural network or convolutional neural network (CNN) for short is a architecture which learns features from data. Proposed methods based on this architecture have recently obtained state-of-the-art results in several applications including texture analysis. This section discusses shortly about a generic CNN, and some well-known CNN because their texture classification results are used for comparison purpose throughout this thesis. They are Lenet, AlexNet, and VGG.

A deep network usually consists of linear operators intertwined with non-linearity ones f which computes a sequence of layers $\Phi_m x$,

$$\Phi_m x = f(W_m ... f(W_2 f(W_1 x))). \tag{4.34}$$

A forward propagation used to form the network layers one after another

$$\Phi_m x = f(W_m \Phi_{m-1} x). \tag{4.35}$$

The last layer $\Phi_M x$ computes a required function $y(x)$ which is the label of the input image for a classification task. The difference between Φ_M and the target function is quantified by an error function $E(\Phi x)$. The derivative of this errors with respect to the values of the output of each layer is calculated by back propagation method [163]. The top error $\frac{\partial E}{\partial \Phi_M}$ is initialized by simply computing the derivative of E in the point $\Phi_M x$, then the other derivatives are obtained recursively by applying the chain rule

$$\frac{\partial E}{\partial \Phi_{m-1} x} = \frac{\partial E}{\partial \Phi_m x} \frac{\partial f(W_m \Phi_{m-1} x)}{\partial \Phi_{m-1} x} \tag{4.36}$$

Once those errors derivatives are computed, the derivatives of the error with respect to the weights are deducted by applying the chain rule

$$\frac{\partial E}{\partial W_m} = \frac{\partial E}{\partial \Phi_m} \frac{\partial f(W_m \Phi_{m-1} x)}{\partial W_m} \tag{4.37}$$

The learning is done typically with a gradient descent algorithm which updates the weights with

$$W_m \leftarrow W_m + \eta \frac{\partial E}{\partial W_m} \tag{4.38}$$

where η is the learning rate.

Convolutional neural networks [164–166] limit the number of weights by localizing the operators W_m and sharing their weights across different positions. Their layers $\Phi_m x(u, p_m)$ are indexed by a spatial variable and a depth variable p_m. The weights are computed by

$$\Phi_m x(x, p_m) = f\left(\sum_{\substack{v \in \Omega \\ p_{m-1}}} \Phi_{m-1}(u, p_{m-1}) w_{v, p_m, p_{m-1}}\right) \tag{4.39}$$

For the localization, the sum is limited to a compact support Ω. However, in the convolution, $w_{v,p_m,p_{m-1}}$ does not depend upon the position u but only upon the offset v and the input and output path p_{m-1} and p_m. In most deep network architecture for images, the spatial resolution of signals (i.e. the number of samples for u) decreases from one layer to the next, while the depth (i.e. the number of samples for p_m) grows, which results in a progressively more rich and more invariant representation. This is similar to translation scattering. A major difference between the translation scattering and convolutional neural network as defined in (4.39) is that in (4.39), every output depth p_m is connected to every input depth p_{m-1}. By contrast, a scattering path $p_m = (\theta_1, j_1, ...\theta_{m-1}, j_{m-1}, \theta_m, j_m)$ is connected to only one previous path, its ancestor $p_{m-1} = (\theta_1, j_1, ...\theta_{m-1}, j_{m-1})$. This implies that the translation invariance is built independently for different path, which can lead to information loss.

The following paragraphs will summarize architecture of three well-known convolutional neural networks which are to some extend related to this thesis.

LeNet-5

LetNet-5 is seven-layer convolutional neural network without counting the input. It was proposed to recognize handwriting digits. Lenet-5's layers contain trainable parameters (weights), ScatNet [35] bears some resemblances to LeNet-5 such as subsampling strategy, convolution layers except this learning functionality.

LeNet-5 has three convolution layers labeled C1, C3, C5 and two subsampling layers labeled S2, S4 as illustrated in Figure 4.5.

Convolutional layer C1 produces six feature maps by using 5×5 filters or 5×5 matrices at stride 1 while convolutional layers C3 and C5 output 16 and 120 feature maps respectively with the same filter size of 5×5.

Sub-sampling layers S2, S4 are applied in the neighborhood of size 2×2 at a

stride of 2. Sigmoid or tanh nonlinearity function, and averaging pooling are used in LeNet-5.

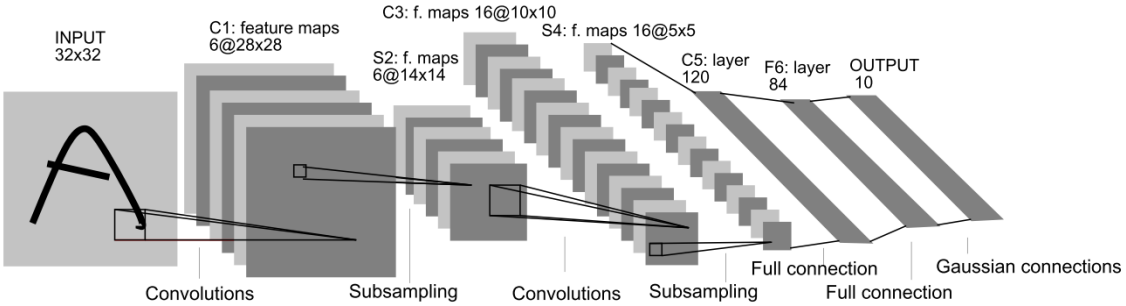


Figure 4.5: Architecture of LeNet-5, a Convolutional Neural Network for digits recognition [67].

AlexNet

The architecture of AlexNet is summarized in Figure 4.6. It contains eight learned layers in which there are five convolutional and three fully-connected ones.

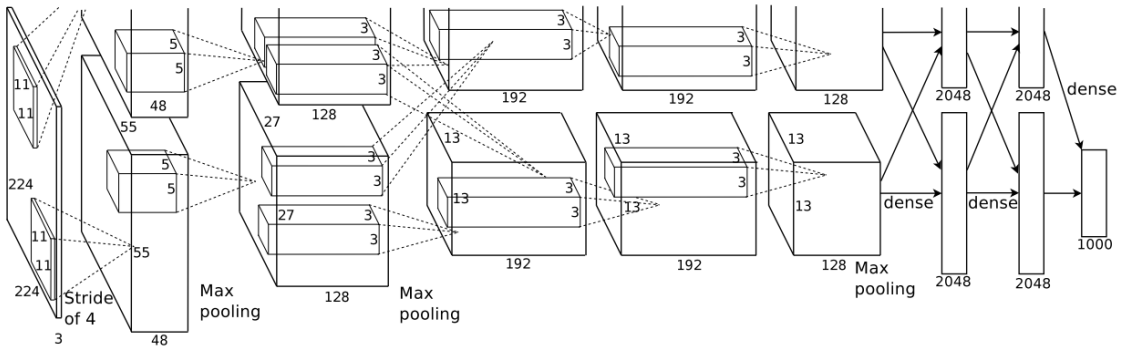


Figure 4.6: An illustration of the AlexNet's architecture, explicitly showing the details of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers [165].

The output of the last fully-connected layer is used as the input of a 1000-way softmax in order to produce a distribution over the 1000 class labels. AlexNet maximizes the multinomial logistic regression objective, which corresponds to maximizing the average across training cases of the log-probability of the correct label under the prediction distribution.

The kernels of the second, fourth, and fifth convolutional layers are connected only to the kernel maps in the previous layer which reside on the same GPU (see Figure 4.6). The kernels of the third convolutional layer are connected to all kernel maps in the second layer. The neurons in the fully-connected layers are connected to all neurons in the previous layer. There are response-normalization layers follow the first and second convolutional layers. Overlapping max-pooling layers, come

after both response-normalization layers as well as the fifth convolutional layer. The Rectified Linear Units (ReLU) non-linearity is applied to the output of every convolutional and fully-connected layer.

The first convolutional layer computes a convolution the $224 \times 224 \times 3$ input image with 96 kernels of size $11 \times 11 \times 3$ with a stride of 4 pixels (this is the distance between the receptive field centers of neighboring neurons in a kernel map). The second convolutional layer takes as input the (response-normalized and pooled) output of the first convolutional layer and filters it with 256 kernels of size $5 \times 5 \times 48$. The third, fourth, and fifth convolutional layers are connected to one another without any intervening pooling or normalization layers. The third convolutional layer has 384 kernels of size $3 \times 3 \times 256$ connected to the (normalized, pooled) outputs of the second convolutional layer. The fourth convolutional layer has 384 kernels of size $3 \times 3 \times 192$, and the fifth convolutional layer has 256 kernels of size $3 \times 3 \times 192$. The fully-connected layers have 4096 neurons each.

VGGNet

VGGNet [12] is a deep convolutional networks for large-scale image recognition, it is developed by K. Simonyan *et al.* from **V**isual **G**eometric **G**roup, University of Oxford, the network is named after the group's name. In VGGNet, small receptive fields with size of 3×3 is used in all convolutional layers with the convolution stride is set to on for building a deeper network compared to prior proposed configurations, the depth is up to 19 weight layers. For example, one 7×7 convolution layer with F feature maps needs $7^2F^2 = 49F^2$ weights while three 3×3 convolution layers need only $3(3^2F^2) = 27F^2$ weights. In one of the VGGNet configurations, 1×1 convolution filters are utilized, which can be seen as a linear transformation of the input channels (followed by non-linearity). Spatial pooling is carried out by five max-pooling layers, which follow some of the convolution layers (not all the convolution layers are followed by max-pooling). Max-pooling is performed over a patch size of 2×2 with the stride of 2. A stack of convolutional layers (which has a different depth in different architectures) is followed by three Fully-Connected (FC) layers: the first two have 4096 channels each, the third performs 1000-way classification and thus contains 1000 channels (one for each class). The final layer is the soft-max layer. All hidden layers are equipped with the rectification ReLU non-linearity.

4.6 Conclusions

This chapter has presented about scattering transform as an image feature extraction method and relevant image techniques which help to implement it. In that way, important concepts have been reviewed. 1) aggregating the coefficients of scattering network results in scattering transform; 2) Scattering network is formed by cascading of consecutive wavelet transform and non-linear modulus operators; 3) wavelet transform is used as an fundamental operator in ScatNet. Finally, because

scattering network is close related to convolutional neural network (CNN), it has been also introduced in this chapter. Our contributions in this thesis are mainly inspired by the two important handcrafted methods, the LBP presented in Chapter 3 and ScatNet in this chapter. Our first contribution based on LBP method will be introduced next chapter.

Chapter 5

Incorporating Information Quantity into Micro LBP-based Features

In this chapter, a statistical approach to static texture representation is developed, which incorporates the complementary information quantity of image intensity into the LBP-based operators. The descriptors, called the completed local entropy binary patterns (CLEBP), capture the distribution of the relationships between statistical measures of image data randomness, calculated over all pixels within a local structure. Without any pre-learning process and any additional parameters to be learned, the CLEBP descriptors convey both global and local information about texture while being robust to external variations. Furthermore, we use biologically-inspired filtering (BF) which simulates the performance of human retina as preprocessing technique. We show that our approach and the conventional LBP have complementary strength and that by combining these algorithms, one obtains better results than either of them considered separately. Experimental results on four large texture databases, including Outex, KTHTIPS2b, CuRet and UIUC show that our approach is more efficient than many state-of-the-arts.

5.1 Introduction

Entropy, as introduced in information theory by Shannon in 1948 [167], offers, in general terms, an indication of randomness in the studied data, has been used in fields such as physics, computer science, statistics, biology, linguistics, neurology, learning and so on.

This chapter introduces a novel texture representation which is built by applying the LBP-based self-similarity operators upon the image entropy space. Different from all existing local binary pattern (LBP) variants, our method encodes the re-

lationships between the statistical measure of the randomness of intensities, i.e., entropy, within a local image structure. Motivated by the completed LBP operators proposed by Guo *et al.* [24], we include both the sign and magnitude components of the difference between the information quantity of a given local image patch and its surrounding counterparts, and the local entropy of the considered patch. In our proposed method, the CLEBP descriptors, without any pre-learning process and any additional parameters to be learned, convey both local and global information about texture. Furthermore, an efficient filtering simulating the fashion how the human retina processes the images observed and extracts their details is presented in order to improve the strength of texture descriptor at the level of preprocessing. Integrating all these ingredients, CLEBP encodes rich descriptive information while having strong robustness against environmental changes. We then show that our approach and the conventional LBP have complementary strength and that by combining these algorithms, one obtains better results than either of them considered separately. Experimental results on four large texture databases show clearly that our approach is more efficient than contemporary ones.

The rest of chapter is organized as follows, Section 5.2 describes CLEBP in detail. We then verify CLEBP in Section 5.3, with extensive experiments on four popular texture datasets and comparisons with various state-of-the-art texture classification techniques. Section 5.4 provides concluding remarks.

5.2 The Completed Local Entropy Binary Patterns (CLEBP)

Different from all existing texture representations, we propose to apply the self-similarity operators on complementary entropy space of images to describe the texture. Entropy, as introduced in information theory by Shannon in 1948 [167], offers, in general terms, an indication of randomness in the studied data, has been used in fields such as physics, computer science, statistics, biology, linguistics, neurology, learning and so on. This section details our new descriptors by incorporating the image intensity randomness into the LBP-based operators, and then presents a bio-inspired filtering which improves the strength of CLEBP descriptors at the level of preprocessing.

5.2.1 Entropy-based texture descriptors

Entropy measures the randomness in the studied data, heterogeneous patches are therefore expected to have higher entropy values than homogeneous patches. The key idea behind our descriptors is to encode the relationships between the randomness of image data of different patches. Details of the entropy descriptors are described as follows.

In the first step, for each pixel in the given image, the entropy of the pixel

intensities within a local structure around the considered pixel is computed and assigned to it. Figure 5.1-(a) illustrates this first step where the entropy e_p of all pixel intensities within the neighborhood is computed and assigned to pixel p. Up to now, in the obtained "map", each pixel conveys the information of its neighborhood. Figure 5.2 illustrates examples of entropy maps which are calculated with square neighborhood of 7×7 pixels (more details about neighborhood are presented in the next section).

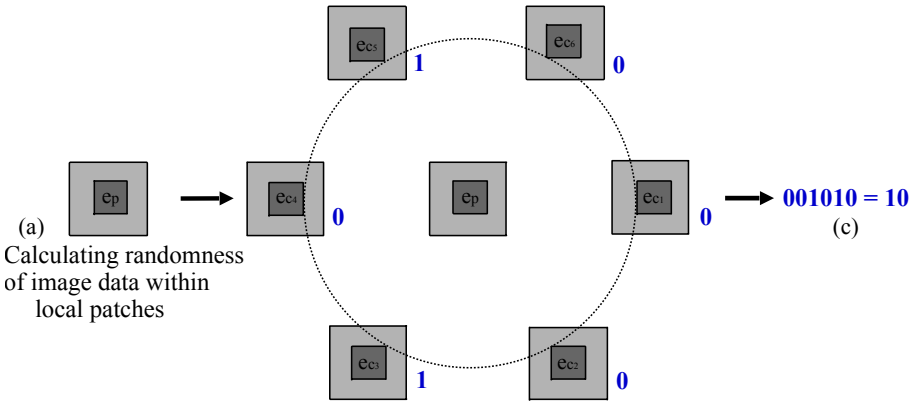


Figure 5.1: Extraction of entropy-based features for a pixel p. e_p represents the entropy value assigned to the considered pixel, e_c denotes the entropy value assigned to the surrounding pixels

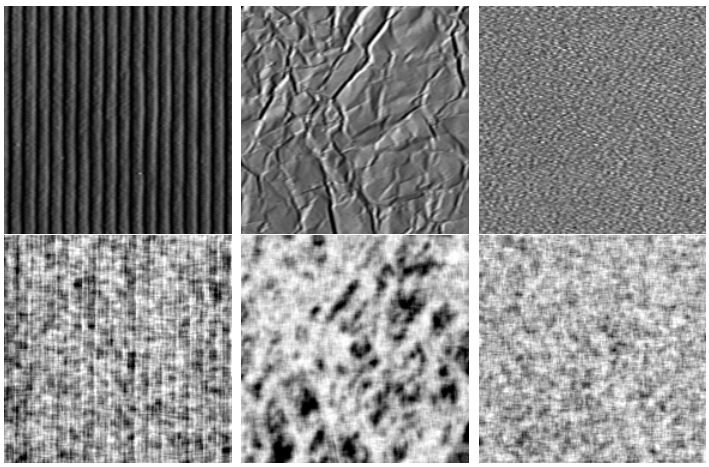


Figure 5.2: Examples of entropy "maps" on which the self-similarity operators are applied. Upper row: original images, lower row: corresponding entropy maps. Our final texture representations are those combining the features extracted from both the upper and lower images, which are intuitively different. We believe therefore that they have complementary strength which is not present in the counterpart.

Then, in the second step, we apply the completed LBP operators on the entropy map to build novel features (example shown in Figure 5.1-(b)). When applying the

CLBP_S operator, we obtain CLEBP_S features (Figure 5.1-(c)):

$$CLEBP\_S_{P,R} = \sum_{p=0}^{P-1} s(e_p - e_c)2^p, \quad s(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases} \tag{5.1}$$

where e_p represents the entropy value assigned to the center pixel (the considered pixel), e_c denotes the entropy value assigned to the surrounding pixels, and the term "E" in CLEBP refers to entropy. Similarly, the CLEBP_M and CLEBP_C features as well as their combination will be obtained (see below for more details about combination strategies).

Since its definition in the seminal work of Shannon [167], entropy has widely used for different fields and can be calculated by different formulas. In this work, various entropy definitions were evaluated, ranging from the basic definition (Shannon entropy) to the generalized ones (Renyi and Tsallis-Havrda-Charvat entropies), as well as the most used in image processing [168–171]. Basically, for each pixel c, the local entropy is calculated in its neighborhood using:

$$e_1(c) = -\sum_{i=1}^{n} p(i) log_2 p(i) \tag{5.2}$$

where n is the total number of different pixel intensities, or gray values, present in the neighborhood and $p(i)$ is the frequency of appearance of value i in the local patch, i.e., the ratio of the number of pixels with value i in the patch to the total number of pixels of the patch.

The Renyi and Tsallis-Havrda-Charvat entropies, both used for image registration [170], are respectively defined by:

$$e_2(c) = \frac{1}{1-\alpha} \sum_{i=1}^{n} log_2 p^{\alpha}(i) \tag{5.3}$$

$$e_3(c) = \frac{1}{\alpha - 1}(1 - \sum_{i=1}^{n} p^{\alpha}(i)) \tag{5.4}$$

where $\alpha > 0$ and $\alpha \neq 1$, and $p(i)$ is the frequency of appearance of value i in the local patch.

Other entropy definitions [168] are also considered:

$$e_4(c) = -\sum_{i=1}^{n} \frac{1}{log_2 p(i)} \tag{5.5}$$

$$e_5(c) = -\sum_{i=1}^{n} \frac{1}{[log_2 p(i)]^2} \tag{5.6}$$

$$e_6(c) = -\sum_{i=1}^{n} \sqrt{log_2 p(i)} \tag{5.7}$$

$$e_7(c) = -\sum_{k \in patch} log_2(\frac{\tilde{I}(k)}{I(k)}) \tag{5.8}$$

$$e_8(c) = -\sum_{k \in patch} \tilde{I}(i) log_2(\frac{\tilde{I}(i)}{I(i)}) \tag{5.9}$$

where, in the two last equations, k is the coordinates of image pixels: all pixels within a patch around the current pixel c are considered; $I(k)$ and $\tilde{I}(k)$ denote respectively the pixel intensities at the pixel k in the original images and its smoothed version.

Properties:

• Entropy, a measure of randomness of data, is invariant by definition to intensity shift. CLEBP descriptors are therefore robust to illumination changes. Note that this robustness comes from both the entropy space and the self-similarity operators whereas other LBP variants use only self-similarity operators.

• Our features are built by two steps, the first step is to characterize the information quantity of image data at a small scale and the second step is to encode the relationship of this information at a larger scale. As a result, our descriptors convey both local and more global information about texture.

• Being built upon the entropy space, we believe therefore that our descriptors have the complementary strength which is not present in CLBP. In practice, the combination approaches (see next section) results in state-of-the-art texture classification performance.

5.2.2 Preprocessing with Biologically Inspired Filtering

This section first briefly describes the human retina, in particular the bipolar cells by which the algorithm is inspired and then details the BF filtering [102].

5.2.2.1 Model of retinal processing

Lying at the back of the eye, the retina is made of three layers: the photoreceptors layer with cones and rods; the outer plexiform layer (OPL) with horizontal, bipolar

and amacrine cells; and the inner plexiform layer (IPL) with ganglion cells [172].

Photoreceptors: rods have the ability to see at night, under conditions of very low illumination; cones have the ability to deal with bright signals. Photoreceptor layer plays therefore the role of a light adaptation filter.

OPL: the photoreceptor performs a low pass filter. Horizontal cells perform a second low pass filter. In OPL, bipolar cells calculate the difference between photoreceptor and horizontal cell responses. Typically, to model the processes of OPL, two Gaussian low pass filters corresponding to the effects of photoreceptors and horizontal cells are used [172]. Thus, bipolar cells act like a difference of Gaussians (DoG) filter.

IPL: IPL works similarly to OPL but it performs on the temporal information rather than on the spatial one as in OPL, and thus does not concern the present work focusing on static images.

5.2.2.2 Details of the BF method

In fact, there are two types of bipolar cells, called ON and OFF. The ON bipolar cells take into account the difference of photoreceptor and horizontal cell responses, whereas the OFF bipolar cells compute the difference of horizontal and photoreceptor cells. If we apply a DoG filter on an image for simulating the bipolar cells, a map with positive and negative values will be obtained. Within this map, the positive values and the absolute of the negatives values correspond respectively to the responses of the ON and OFF cells. The DoG filter calculates the second spatial derivative of an image. In areas where the image has a constant intensity, the filter response will be zero. Wherever an intensity change occurs, the filter will give a positive response on the darker side and a negative response on the lighter side. To simulate the performance of the ON and OFF cells, a two-step preprocessing technique is presented as follows (Figure 5.3):

Step 1: the given image I_{in} is first filtered by a band-pass DoG filter: $I_{bf} = DoG * I_{in}, DoG = \frac{1}{2\pi\sigma_1^2} e^{-\frac{x^2+y^2}{2\sigma_1^2}} - \frac{1}{2\pi\sigma_2^2} e^{-\frac{x^2+y^2}{2\sigma_2^2}}$ where σ_1 and σ_2 correspond to the standard deviations of the low pass filters modeling photoreceptors and horizontal cells.

Step 2: the responses at bipolar cells are then decomposed into two maps corresponding to the image details alongside the two sides of the image edge:

$$I_{bf}^{+}(p) = \begin{cases} I_{bf}(p) & if \quad I_{bf}(p) \geq \epsilon \\ 0 & otherwise \end{cases} \tag{5.10}$$

$$I_{bf}^{-}(p) = \begin{cases} |I_{bf}(p)| & if \quad I_{bf}(p) \leq -\epsilon \\ 0 & otherwise \end{cases} \tag{5.11}$$

where the term bf refers to "Biologically-inspired Filtering", p refers to a considered

pixel, ϵ is slightly larger than zero to provide some stability in uniform regions: we do not take into account the uniform areas since these areas often contain noise rather than useful texture information.

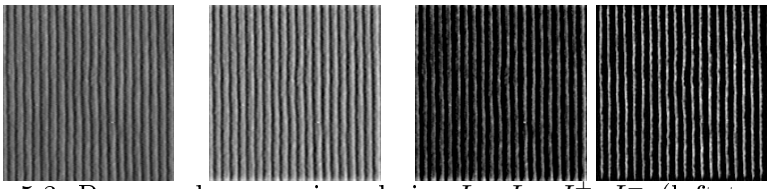


Figure 5.3: Proposed processing chain: $I_{in}, I_{bf}, I_{bf}^{+}, I_{bf}^{-}$ (left to right).

Then, in the feature extraction step, instead of the input image I_{in}, features are first extracted from the two images, I_{bf}^{-} and I_{bf}^{+}, and then combined together. The resultant feature vector is considered as the descriptor of I_{in}.

As advantages, the BF filter is first robust to illumination and noise since the band-pass DoG filter removes both high frequency noise and low frequency illumination. Also, BF is isotropic and discards all orientation information, and is therefore independent to rotation. The BF filter is real time: with un-optimized MATLAB code running on a laptop of CPU Intel Core i5 1.7 Ghz (2G Ram), it takes less than 0.9 s to process 1000 images of 128 $\times$ 128 pixels.

5.3 Experimental validation

5.3.1 Experimental settings

Databases

The effectiveness of the CLEBP is assessed by a series of experiments on four large and representative databases: Outex [9], CUReT [47], UIUC [52] and KTHTIPS2b [173].

The Outex database (examples are shown in Figure 2.7) contains textural images captured from a wide variety of real materials. We consider the three commonly used test suites (TC10, TC12 t184, and TC12 horizon) containing 24 classes of textures which were collected under three different illuminations ("horizon", "inca", and "t184") and nine different rotation angles (0°, 5°, 10°, 15°, 30°, 45°, 60°, 75° and 90°).

The CUReT database (examples are shown in Figure 2.4) contains 61 texture classes, each having 205 images acquired at different viewpoints and illumination orientations. There are 118 images shot from a viewing angle of less than 60°. From these 118 images, as in [24, 174], we selected 92 images, from which a sufficiently large region could be cropped (200×200) across all texture classes. All the cropped regions are converted to grey scale. The UIUC texture database includes 25 classes with 40 images in each class. The resolution of each image is 640×480. The database con-

tains materials imaged under significant viewpoint variations (examples are shown in Figure 2.8).

The acquisition procedure for KTH-TIPS-2b has been described in more detail in [173], with 3 viewing angles, 4 illuminants, and 9 different scales, producing 432 images per class. Figure 2.6 illustrates an example of the 11 materials. All the images are converted to grey scale. Notice in particular the striking differences between samples of the same class. There is almost no intra-class variation due to in-plane rotation for this database.

Combination strategies, similarity measure, and classifier

There are two ways to combine two types of codes, i.e., the CLEBP_S and CLEBP_M codes [1] or CLBP_S and CLBP_M ones, concatenation or jointly [24]:

- In the first way, the histograms of the LEP_S and LEP_M codes are computed separately, and then concatenated together. This scheme is referred to as "LEP_S_M" (CLBP_S_M resp.).
- In the second way, a joint 2D histogram of the LEP_S and LEP_M codes is calculated and denoted by "LEP_S/M".

We consider now how to combine the three operators, LEP_S, LEP_M and LEP_C [2]. Similarly, they can be combined in two ways, jointly or hybridly:

- In the first way, a 3D joint histogram of them is built and denoted by "LEP_S/M/C".
- In the second way, a 2D joint histogram, "LEP_S/C" or "LEP_M/C" is built first, and then is converted to a 1D histogram, which is then concatenated with LEP_M or LEP_S to generate a joint histogram, denoted by "LEP_M_S/C" or "LEP_S_M/C".

All those combination strategies were evaluated in our experiments. Similar to [24], we found that the "joint" combination strategies outperform the others. We therefore present in the next section the results of joint strategies which are denoted by "LEP_SM, "LEP_SMC","CLBP_SM", and "CLBP_SMC", for short. Concerning how to combine two models CLEBP and CLBP, we consider only the concatenation strategy in order to avoid the high complexity (the complexity of the joint strategies is higher than that of concatenation ones). Classification rates are reported using the simple nearest neighbor classifier (NNC).

[1] For a better distinction between CLEBP and CLBP, the CLEBP is simply denoted by LEP hereafter.

[2] Throughout experiments carried out in this chapter, only the rotation invariant uniform patterns are considered. The rotation invariant uniform patterns are often denoted with the subscript ^{riu_2}, e.g., LEP_S^{riu_2}, but for simplicity in this chapter, they are simply denoted by their names, without ^{riu_2}, e.g., LEP_S.

5.3.2 LEP descriptor settings

Parameters of the LEP descriptors include the entropy definition, the shape and size of the neighborhood where entropy is calculated, as well as the quantization. We first evaluated the performance of the proposed descriptors with respect to different entropy definitions, as mentioned in Section 5.2.1. In our experiments on the Outex database, we found that they perform similarly (indeed, the classic Shannon entropy functions slightly better than the others). Note that computing entropy is just the first step in our algorithm and we further encode the relationships of local entropy of different patch. That is the reason why the performance of our descriptors is somehow independent of the entropy definitions. In the rest of this chapter, we use the classic Shannon entropy for its computational efficiency.

Concerning the shape of neighborhood, two cases are considered: square (grid) and circle (Figure 5.4). In the first case, the intensities of all pixels within the image grid around the considered pixel are used whereas in the second case, the intensities of neighbors in several circles are used. In the second case, interpolation is used for estimating the neighbors which do not fall in the center of pixels. Theoretically, the circle neighborhoods should perform better than the square ones since circle-sampling neighbors with interpolation makes the entropy computation invariant to rotation. To validate this, we computed the texture classification rates on three test suites (TC10, TC12h and TC12t) of the non-preprocessing Outex texture database [9] for the two schemes. When using the LEP_S operator, the circle neighborhoods outperform the square ones by 4-5% but when combining the LEP_S, LEP_M and LEP_C operators, the two types of neighborhood perform similarly. We use therefore the square neighborhoods for their low complexities.

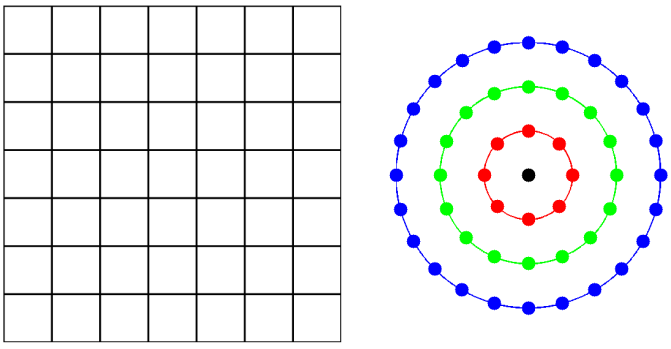


Figure 5.4: Entropy can be computed using all pixels within the image grid (left) or the neighbors in one or many circles around the considered pixel (right).

Concerning the dimension of neighborhood, a trade-off needs to be considered: on the one hand, the neighborhood needs to be rather small in order to increase spatial resolution and capture local details in the texture. On the other hand, in order to have reliable and meaningful statistical analysis, the number of considered values has to increase or the window size increases. By experiments on the Outex database, we obtained the best performance when using the CLEBP descriptors built upon the entropy space calculated within square neighborhood of 7×7 pixels

(neighborhoods of 3×3, 5×5, 7×7, and 9×9 are considered).

Two quantization strategies were tested, one with the entire image being quantized to q gray levels (q is a parameter) and one with each considered patch being quantized individually. Another particular case, referred to as the third strategy hereafter, is that no quantization is used, i.e., $q = 256$. Regarding the statistical analysis, the neighborhood size and the quantization are relatively dependent: when the neighborhood size increases, the number of gray levels quantized q should increase. When q increases, the local variance of the patch is more precisely characterized, the performance of a considered strategy increases. Similarly, with the same q, the second quantization strategy should perform better the first one. However, the higher q is, the higher the computational cost of entropy is. The second quantization strategy is slower than the first one. In our experiments on the Outex database when using the LEP_S operator with $q = 8, 16, 32$, the second quantization strategy outperforms the first one by 2-3%, but the computational time is of around 70% higher. The simple third strategy without quantization results in the similar performance as the second strategy with $q = 32$ with the computational time of 20% lower. In the follows, we report the results of our descriptors without quantization.

5.3.3 BF experimental settings

This section explores different parameters of BF preprocessing algorithm to improve the performance of LBP features for texture classification.

Feature extraction

Concerning LBP, we use the completed model proposed in [24], which gathers three individual operators: CLBP-Sign (CLBP_S) which is equivalent to the conventional LBP, CLBP-Magnitude (CLBP_M) which measures the local variance of magnitude, and CLBP-Center (CLBP_C) which extracts the local central information. When using LBP features, only rotation invariant uniform patterns are considered [3].

We also apply the combination strategies presented in [24]. There are two ways to combine the CLBP_S and CLBP_M codes: by concatenation or jointly. In the first way, the histograms of the CLBP_S and CLBP_M codes are computed separately, and then concatenated together. This CLBP scheme is referred to as "CLBP_S_M". In the second way, a joint 2D histogram of CLBP_S and CLBP_M codes is calculated (denoted as CLBP_S/M).

Finally, the three operators, CLBP_S, CLBP_M and CLBP_C, can be combined in two ways, jointly or hybridly. In the first way, a 3D joint histogram of them

[3] The rotation invariant uniform patterns are often denoted with the subscript ^{riu_2}, e.g., LBP^{riu_2} or LBC^{riu_2}, but for simplicity they are simply denoted by their names, without ^{riu_2}, e.g., LBP

is built and denoted by "CLBP_S/M/C". In the second way, a 2D joint histogram, "CLBP_S/C" or "CLBP_M/C" is built first, and then is converted to a 1D histogram, which is then concatenated with CLBP_M or CLBP_S to generate a joint histogram, denoted by "CLBP_M_S/C" or "CLBP_S_M/C".

We therefore apply these schemes on the original texture images and on the preprocessed images, obtained as presented in Section 5.2.2.

Similarity measure and classifier

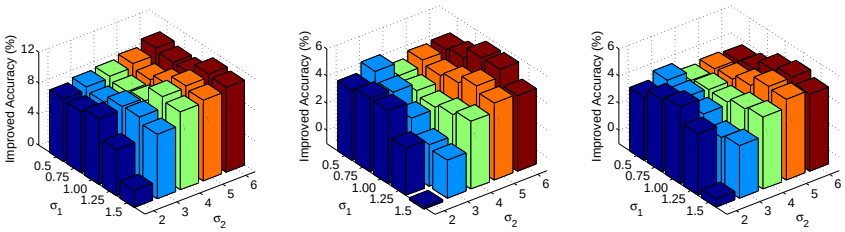
Classification rates are reported using the simple nearest neighbor classifier. The χ^2 distance is used to measure the similarity between two texture images. If $H = \{h_i\}$ and $K = \{k_i\}, (i = 1, 2, ..., n)$ denote two histograms corresponding to the representations of two images, the χ^2 distance is calculated as:

$$\chi^2(H, K) = \sum_{i=1}^{n} \frac{(h_i - k_i)^2}{h_i + k_i} \tag{5.12}$$

5.3.3.1 BF parameter exploration

In this section, we study how the parameters of the BF filter its influence final performance. Parameters to be chosen include the two standard deviations σ_1 and σ_2 defining the low and high cutoff frequencies of the band pass *DoG* filter, and the threshold ϵ. A critical constraint is $\sigma_1 < \sigma_2$.

The experiments described in this section were conducted on the Outex database. We compute the average classification rates on the three test suites (TC10, TC12t and TC12h) with different parameters of the BF filter. For each texture descriptor or combination strategy in the completed LBP model used (each descriptor was also tested with different parameters itself), 150 BF filters with different parameters were evaluated: $\sigma_1 \in \{0.5, 0.75, 1, 1.25, 1.5\}$, $\sigma_2 \in \{2, 3, 4, 5, 6\}$, $\epsilon \in \{0.05, 0.1, 0.15, 0.2, 0.25, 0.3\}$.



(a) CLBP_S/C$_{R=1}$ (b) CLBP_M/C$_{R=2}$ (c) CLBP_S_M/C$_{R=2}$

Figure 5.5: Improved classification accuracy obtained with different standard deviations (σ_1 and σ_2) of the BF filter. These rates are calculated by averaging classification rates with different ϵ on the three test suites (TC_10, TC_12 "t", TC_12 "h") of the Outex database.

• Determining σ_1 and σ_2

For each pair of σ_1 and σ_2, the average classification rates are calculated for varying ϵ. Figure 5.5 shows the *improved* classification rates obtained when BF filter is used with different descriptors and combination strategies. In general, Vu *et al.* [102] experimentally recommended to use $\sigma_1 \in \{0.7, 1.3\}$ and $\sigma_2 \in \{3, 6\}$. Indeed, in our tests, we obtained the best results with $\sigma_1 = 1.25$ and $\sigma_2 = 5, 6$.

• Determining ϵ

The slightly larger than zero threshold parameter ϵ is used to provide the stability in uniform regions (the uniform areas containing noise rather than useful texture information should be removed). To determine ϵ, the average of classification rates across different ϵ is computed: for each value of ϵ, 12 pairs of (σ_1, σ_2), $\sigma_1 = 0.75, 1.00, 1.25$ and $\sigma_2 = 3, 4, 5, 6$ are used. As can be seen from Figure 5.6, the performance of the BF filter with $\epsilon = 0.05, 0.1, 0.15$ is similar (the BF filter with $\epsilon = 0.15$ performs slightly better the others) while it begins to drop when $\epsilon \geq 0.2$. This can be explained by the fact that if ϵ is too high, useful information is lost.

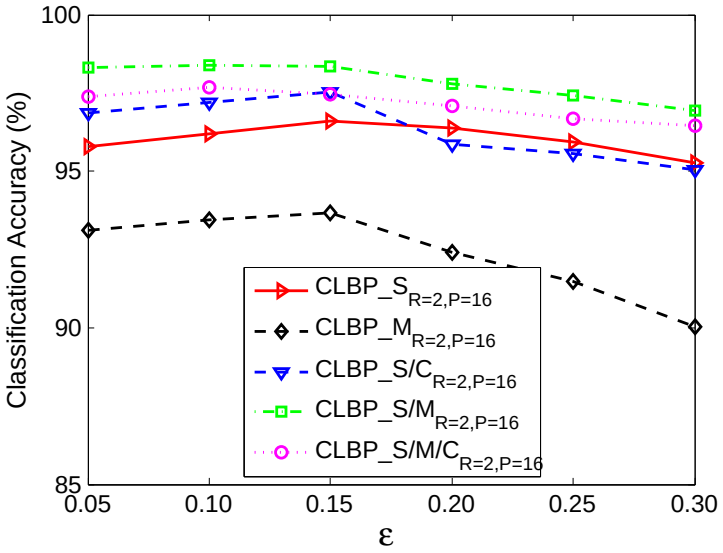


Figure 5.6: Classification rates for different descriptors with different values of the threshold ϵ. These rates are calculated by averaging classification rates on the three test suites (TC_10, TC_12 "t", TC_12 "h") of the Outex database with different values of σ_1 and σ_2.

To summarize, the BF filtering performs well with different parameters: $\sigma_1 \in [0.7, 1.3]$ and $\sigma_2 \in [3, 6]$, $\epsilon \in [0.05, 0.15]$. The best results are obtained with $\sigma_1 = 1.25$, $\sigma_2 = 5, 6$, and $\epsilon = 0.15$ when CLBP is used.

5.3.3.2 The robustness of BF to noise

Robustness to noise is one of the most important factors to assess texture classification methods. To measure the robustness of the BF we use the three test suites of the Outex dataset. In our experiments, the original texture images are added with random Gaussian noise with different signal-to-noise ratios (SNR): SNR = {30, 15, 10, 5, 4, 3}. To reduce the variability of randomness, each experiment is repeated ten times and then the average classification accuracies and the standard deviations are calculated for each test suite and for each descriptor or combination strategy.

Table 5.1: Classification rates on the Outex test suites with different signal-to-noise.

		SNR=30	SNR=15	SNR=10	SNR=5	SNR=4	SNR=3
TC10	CLBP_S/M$_{R=2}$	97.87	97.45	96.70	91.01	88.99	86.81
	BF + CLBP_S/M$_{R=2}$	99.26	99.27	99.17	98.76	98.52	98.04
TC12t	CLBP_S/M$_{R=2}$	90.44	89.71	88.01	82.61	80.02	78.38
	BF + CLBP_S/M$_{R=2}$	98.16	98.07	97.82	97.08	96.95	96.44
TC12h	CLBP_S/M$_{R=2}$	91.07	90.51	88.41	82.71	81.02	78.03
	BF + CLBP_S/M$_{R=2}$	97.75	97.64	97.51	97.27	96.99	96.21

As can be seen from Table 5.1, the BF method is robust to noise. When the BF preprocessing is used, the classification accuracy is much more stable for the evaluated descriptor. This loss is about 5% for the combination strategies such as CLBP_S/M and CLBC_S/M, and more than 12% for the "individual" descriptor like LBP. When the BF filter is applied, for CLBP_S/M and CLBC_S/M, the losses are around 0.5% with SNR=5 for all three test suites.

5.3.3.3 Low complexity

Concerning the complexity of an algorithm, three factors must be considered: preprocessing time, feature extraction time and feature size which affects the classification time. BF preprocessing step requires $0.87ms$ (resp. $1.91ms$) to process an image of 128×128 (resp. 200×200) pixels, this additional time being really small. When using the BF filter, the feature extraction time and feature size are doubled, but there are important gains in classification rates. The feature extraction time of "BF + LBP" is similar to that of CLBP_S_M/C and CLBP_S/M. However, while the feature sizes of CLBP_S_M/C are 30, 54, and 78, the feature sizes of CLBP_S/M are 100, 324, and 676 for $(R = 1, P = 8)$, $(R = 2, P = 16)$, and $(R = 3, P = 24)$, respectively, the feature sizes of "BF + LBP" are much smaller: they are 20, 36, and 52 for $(R = 1, P = 8)$, $(R = 2, P = 16)$, and $(R = 3, P = 24)$ respectively. This means that "BF + LBP" is more efficient than CLBP_S_M/C and CLBP_S/M in both terms of performance and complexity.

5.3.4 Results on the Outex database

In our LEP descriptors, the first step is computing entropy map in which a pixel conveys the information of local patch around it. In the second step, the relationship between the entropy of neighbor patches is encoded. If R is small, considered patches will significantly overlapped, thus, in the experiments on the Outex database, we report the performance of the LEP descriptors with $R = 3$ and varying P ($P = 8, 16$, or 24). Given an image, the BF filtering is first applied and the two maps are obtained (see I_{bf}^{+} and I_{bf}^{-} in Figure 5.3). The LEP_SMC and CLBP_SMC features are then extracted on both I_{bf}^{+} and I_{bf}^{-}. All the four resulting descriptors are then concatenated in order to obtain the final descriptor representing the considered texture image. Recall that, in the three tests of this database, the training and testing images are priorly split by its predefined evaluation protocol.

Table 5.2 compares the performance obtained by combining the BF, LEP_SMC and CLBP_SMC to that of different recent state-of-the-art systems. Our approach outperforms significantly and systematically all competing algorithms. The best results of the NI/RD/CI (Neighboring Intensities, Radial Difference, and Central Intensity) [74] which are respectively 99.7%, 98.7% and 98.1% for the three test suites of the Outex database are obtained by combining the three descriptors at multi-scale while our results, 99.9, 99.3, and 99.6% on the three tests respectively, are obtained with *single-scale features* ($CLBP_{(R,P)=(2,16)} + LEP_{(R,P)=(3,16)}$). Also, in [33], in order to obtain the results of 99.3, 97.7, and 98.6%, the authors have to combine *nine* scales. TDLBP&NGF [175] are fused features of the DLBP features and the normalized Gabor filter response average magnitudes (NGF). The DLBP approach needs pre-training and the dimensionality of the DLBP feature varies with the training images. The results of CLBP_CLBC included are the best ones obtained by combining CLBP_S/M/C and CLBC_S/M/C with $(R, P) = (3, 24)$ [133]. To the best of our knowledge, our classification rates obtained are the best ever published results on this database.

Table 5.2: Comparing CLEBP with various state-of-the-art methods on the Outex database.

	TC10	TC12		Mean
		t184	horizon	
CLEBP	99.88	99.26	99.56	99.57
VZ-MR8 [176]	93.39	92.55	92.82	92.99
VZ-Joint [174]	92.00	91.41	92.06	91.82
LBPV [72]	97.63	95.06	93.88	95.52
CLBP [24]	99.14	95.18	95.55	96.62
LBPV [72]	97.63	95.06	93.88	95.52
CLBC [133]	98.96	95.37	94.72	96.35
CLBP_CLBC [133]	98.96	95.37	94.72	96.35
NI/RD/CI [74]	99.70	98.70	98.10	98.83
BRINT [33]	99.35	97.69	98.56	98.53
MRELBP [34]	99.38	99.17	99.51	99.35

We now discuss in more detail the contribution of each algorithm presented, LEP and BF, to those excellent obtained results. The detailed results are reported

Table 5.3: Classification rates obtained on the Outex database.

		TC10	TC12		Mean
			t184	horizon	
R=1, P=8	CLBP_SM	94.66	82.75	83.14	86.85
	BF + CLBP_SM	97.18	95.01	94.00	95.39
	CLBP_SMC	96.56	90.30	92.29	93.05
	BF + CLBP_SMC	97.40	95.46	94.97	95.94
	LEP_SM	90.11	83.26	81.64	85.00
	BF + LEP_SM	94.14	91.85	93.45	93.15
	LEP_SMC	91.23	85.20	83.81	86.75
	BF + LEP_SMC	97.79	96.13	96.85	96.92
	LEP_SM + CLBP_SM	97.79	91.83	91.46	93.69
	BF + LEP_SM + CLBP_SM	97.86	97.36	97.82	97.68
	LEP_SMC + CLBP_SMC	98.36	94.42	95.23	96.00
	BF + LEP_SMC + CLBP_SMC	**99.14**	**98.52**	**98.77**	**98.81**
R=2, P=16	CLBP_SM	97.89	90.55	91.11	93.18
	BF + CLBP_SM	99.61	97.82	98.84	98.66
	CLBP_SMC	98.72	93.54	93.91	95.39
	BF + CLBP_SMC	99.53	97.69	98.73	98.65
	LEP_SM	93.90	84.70	87.15	88.58
	BF + LEP_SM	97.34	94.84	96.00	96.06
	LEP_SMC	94.43	86.39	90.92	90.58
	BF + LEP_SMC	99.11	97.66	98.22	98.33
	LEP_SM + CLBP_SM	99.09	95.60	95.20	96.63
	BF + LEP_SM + CLBP_SM	99.79	98.98	99.44	99.40
	LEP_SMC + CLBP_SMC	99.19	96.60	96.11	97.30
	BF + LEP_SMC + CLBP_SMC	**99.88**	**99.26**	**99.56**	**99.57**
R=3, P=24	CLBP_SM	99.32	93.58	93.35	95.41
	BF + CLBP_SM	99.66	98.35	99.03	99.01
	CLBP_SMC	98.93	95.32	94.53	96.26
	BF + CLBP_SMC	99.66	98.45	99.00	99.04
	LEP_SM	94.56	87.92	86.32	89.60
	BF + LEP_SM	97.65	94.97	95.87	96.16
	LEP_SMC	94.32	88.75	87.46	90.18
	BF + LEP_SMC	99.08	97.63	97.82	98.17
	LEP_SM + CLBP_SM	99.09	96.55	96.16	97.27
	BF + LEP_SM + CLBP_SM	99.90	98.89	99.47	99.42
	LEP_SMC + CLBP_SMC	99.19	97.22	96.32	97.58
	BF + LEP_SMC + CLBP_SMC	**99.82**	**99.26**	**99.56**	**99.55**

in Table 5.3 (the bold lines indicate the best final results - "mean" column) from which we can get some interesting findings:

• For considered features with their different parameter settings, using BF as preprocessing results in high gains in performance. For example, when combining BF with CLBP_SM with the three parameter configuration, the average improvements are respectively 8.54%, 5.48%, and 3.6%.

• The BF filter and the LEP descriptors have the good complementary strengths. The LEP features are less powerful than CLBP but the combination of BF and LEP performs similarly well the combination of BF and CLBP. For example, with $R = 1$, the classification rates of "BF + LEP_SMC" and "BF + CLBP_SMC" are respectively 96.92% and 95.95%. With $R = 2$, the classification rates of "BF + LEP_SMC" and "BF + CLBP_SMC" are respectively 98.33% and 98.65%. It is worth noting that the very high performance of *single scale* "BF + LEP_SMC" ($R = 2$) is already comparable to the state-of-the-art (see Table 5.2): *single scale* "BF + LEP_SMC" results in the similar classification rate of the combination of three descriptors at multi-scale NI/RD/CI [74].

• When two methods BF and LEP combine with the existing CLBP, the results obtained are even better for all considered parameter settings. While CLBP considers the relationships between image intensities, our LEP method captures the relationships between statistical measures of image data randomness. Moreover, the LEP descriptors are built by two steps, the first step is to characterize the information quantity of image data at a small scale and the second step is to encode the relationship of this information at a larger scale. As a result, the LEP descriptors convey both local and more global information about texture. That is the reason why our descriptors have the complementary strength which is not present in CLBP, and the combination of BF, LEP, and CLBP results in excellent classification rates.

5.3.5 Results on the CUReT and UIUC databases

Since the experiment settings as well as the observations obtained on those two datasets are similar, we present the results on those databases in one section. In the experiments on the CUReT database, as in [24], to get statistically significant experimental results, N training images were randomly chosen from each class while the remaining $92 - N$ images per class were used as the test set. Similarly, in experiments on the UIUC database, to eliminate the dependence of the results on the particular training images used, N training images were randomly chosen from each class while the remaining $40 - N$ images per class were used as test set.

Similar to experiments on the Outex database, we first compare CLEBP with the state of the art methods. We include different features: VZ-MR8, LBP and its variants, such as LBP/VAR, DLBP (Dominant LBP), multi-scale CLBP_SMC, multi-scale CLBC_SMC, multi-scale NI/RD/CI, as well as the two very recent algo-

Table 5.4: Comparing CLEBP with state-of-the-art methods on the CUReT et UIUC databases. Our results are obtained by the combination of BF, single-scale LEP, and single-scale CLBP. The bracketed numbers are the number of the training samples per class used for the corresponding database.

Method	CUReT (46)	UIUC (20)
Ours	97.92	95.78
LBP/VAR	55.49	-
DLBP [175]	84.93	60.73
VZ_MR8 [176]	97.79	-
Multi-scale CLBP_SMC[24]	97.40	93.26
Multi-scale CLBC_SMC[133]	95.39	92.42
Multi-scale NI/RD/CI[74]	97.29	-
MRELBP [34]	97.10	-
SSLBP (NNC) [177]	98.55	97.02

Multi-scale methods must combine three scales R.

rithms, MRELBP [34] and SSLBP (Selective Scale LBP) [177]. For a fair comparison, all included results are obtained with a NNC classifier. The average classification rates on those databases over a hundred random splits with different parameters (in those experiments, $R_{LEP} = R_{CLBP}+2$) are reported in Table 5.4, from which we can see the efficiency of our approach. When combining BF, single-scale LEP and CLBP (with $P = 16$), the obtained results are better than almost competing methods, such as multi-scale CLBP_SMC, multi-scale CLBC_SMC, multi-scale NI/RD/CI, and MRELBP. Our method performs only worse than the recent SSLBP method [177] which is based on dominant LBP in scale space and has a strategy for selecting the texture scale.

Tables 5.5 and 5.6 present in more detail the results obtained on the CUReT and UIUC databases, from which we get the similar interesting observation as in the experiments on the Outex database: the BF filter and the LEP descriptors have the complementary strengths (when combining BF and LEP, we already obtained the very high performance, comparable to the state-of-the-art) but when these two methods combine with the existing CLBP, the results obtained are even better.

5.3.6 Results on the KTHTIPS2b database

For the experiments on KTHTIPS2b, we follow the training and testing scheme used in [173]. We perform experiments training on one, two, or three samples; testing is always conducted only on un-seen samples. For example, there are six experiments where two samples are used as training and the two remaining samples are used as testing, and the classification rate reported is the average of those results . In this last database, in order to emphasis the strength of the proposed features, any preprocessing is used. Table 5.7 details the our results obtained on this challenging dataset with different parameters and combination strategies while Table 5.8 compares CLEBP with various state-of-the-art methods which are recently published. As can be seen, our approach considerably outperforms almost all competing algorithms (we obtained the similar results as those of MRELBP). The best results of

Table 5.5: Classification rates obtained on the CUReT database.

		46	23	12	6
R=1, P=8	CLBP_SM	93.52	88.67	81.95	72.30
	CLBP_SMC	95.59	91.35	84.92	74.80
	BF + LEP_SM	92.95	87.91	81.47	72.03
	BF + LEP_SMC	95.13	90.87	83.79	74.67
	BF + LEP_SM + CLBP_SM	96.98	93.96	89.03	83.07
	BF + LEP_SMC + CLBP_SMC	96.92	94.06	89.53	82.58
R=3, P=16	CLBP_SM	94.45	90.40	84.17	75.39
	CLBP_SMC	95.86	92.13	86.15	77.04
	BF + LEP_SM	92.87	88.03	82.01	73.15
	BF + LEP_SMC	94.93	90.68	84.45	74.56
	BF + LEP_SM + CLBP_SM	**97.92**	**95.27**	**91.85**	**83.95**
	BF + LEP_SMC + CLBP_SMC	97.61	94.87	89.95	82.88
R=5, P=24	CLBP_SM	93.63	89.14	82.47	73.26
	CLBP_SMC	94.74	90.33	83.82	74.46
	BF + LEP_SM	92.65	88.61	81.89	73.43
	BF + LEP_SMC	94.78	91.21	84.32	74.48
	BF + LEP_SM + CLBP_SM	97.46	94.78	90.97	82.77
	BF + LEP_SMC + CLBP_SMC	96.61	93.45	88.94	81.83

Table 5.6: Classification rates obtained on the UIUC database.

		20	15	10	5
R=2, P=16	CLBP_SM	87.87	85.07	80.59	71.64
	CLBP_SMC	91.04	89.42	86.29	78.57
	BF + LEP_SM	93.25	89.32	85.47	73.53
	BF + LEP_SMC	93.43	89.27	86.01	75.12
	BF + LEP_SM + CLBP_SM	**95.78**	**92.36**	**90.19**	**81.12**
	BF + LEP_SMC + CLBP_SMC	95.42	92.27	89.97	80.76
R=3, P=24	CLBP_SM	89.18	87.42	81.95	72.53
	CLBP_SMC	91.19	89.21	85.95	78.05
	BF + LEP_SM	92.95	87.91	81.47	72.03
	BF + LEP_SMC	95.13	90.87	83.79	74.67
	BF + LEP_SM + CLBP_SM	95.68	91.96	89.03	81.07
	BF + LEP_SMC + CLBP_SMC	95.32	92.06	89.83	80.78

the NI/RD/CI with training on one, two and three samples are 58.1%, 62.9%, and 66.0% respectively while our results are 58.8%, 65.3%, and 69.1%. It is worth noting that the results of the combination of single-scale features are already better than those of many existing methods (refer to the results of "LEP_SM + CLBP_SM" with $P = 8$ and training on two and three samples which are 64.1% and 67.6% respectively).

Table 5.7: Classification rates obtained on the KTHTIPS2b database.

P	8			16			24		
N_{train}	1	2	3	1	2	3	1	2	3
CLBP_S	48.1	54.2	52.6	50.5	55.8	59.1	49.9	54.6	57.8
LEP_S	48.04	54.15	56.64	49.01	53.75	56.73	48.51	55.03	55.22
LEP_SM	54.60	59.53	61.57	53.63	58.97	61.97	53.61	59.49	63.26
LEP_SMC	55.60	59.77	62.27	53.21	58.45	62.04	51.92	57.02	60.16
LEP_SM + CLBP_SM	58.12	64.11	67.59	57.85	63.98	67.53	58.45	64.01	67.57

P	(8)+(16)			(16)+(24)			(8)+(16)+(24)		
N_{train}	1	2	3	1	2	3	1	2	3
LEP_SM + CLBP_SM	**58.78**	**65.25**	**69.13**	57.93	63.86	67.38	58.84	65.17	68.83

Table 5.8: Comparing CLEBP with various state-of-the-art methods on the KTHTIPS2b database. All scores are obtained with NNC classification.

Method	Accuracy (%)
Ours	69.13
LBP [69]	60.35
CLBP [24]	63.87
PRICoLBP [135]	61.17
MWLD [178]	64.7
VZ_MR8 [176]	55.7
VZ_Joint [174]	60.7
NI/RD/CI [74]	64.84
MRELBP [34]	69.13

5.4 Conclusions

Novel efficient texture features are built by applying the LBP operators upon the image entropy map where each pixel is replaced by the randomness of image intensities within a local patch around it. CLEBP descriptors, without any pre-learning process and any additional parameters to be learned, convey both global and local information about texture. A biologically-inspired filtering (BF) simulating the performance of human retina is also presented as preprocessing. Integrating all these ingredients, CLEBP encodes rich descriptive information while reserving the robustness against environmental changes. Experiments carried out on four large and challenging databases, including Outex, KTHTIPS2b, CuRet, and UIUC provide the better results than many state-of-the-art approaches. Next chapter will introduce an alternative approach texture representation, a compound framework formed by combining two complementary types of features results in high discriminative texture method.

Chapter 6

LBP-ScatNet: An Complementary micro-macro Approach

The previous chapter introduces a novel LBP variant which leverages the information quantity to compensate for the loss of macro-structure information of LBP operator. This results in a robust LBP variant for texture representation. In this chapter, we propose a micro-macro feature combination approach for texture classification. By this way, Local Binary Pattern (LBP) plays the role of micro-structure feature extractor while the scattering transform captures macro-structure information[1]. In fact, for extracting the macro-type features, coefficients from three different layers of the scattering network are aggregated. The scattering network is a handcrafted convolution network which is implemented by computing consecutively wavelet transforms and modulus non-linear operators. By contrast, in order to extract micro-structure features which are rotation-invariant, relatively robust to noise and illumination change, the completed LBP is utilized alongside the biologically-inspired filtering (BF) preprocessing technique. Overall, since the proposed framework can exploit the advantages of both feature types, its texture representation is not only invariant to rotation, scaling, illumination change but also highly discriminative. Intensive experiments conducted on many texture benchmarks such as CUReT, UIUC, KTH-TIPS-2b, and OUTEX show that our framework has a competitive classification accuracy. We discuss and present reasons as to why the two disparate yet complementary categories of features, the scattering transform and

[1]Macro-structure information in the scattering transform is different from those of CLEBP presented in Chapter 5 because of twofold. 1) scattering transform or ScatNet extracts macro features from the input image space based on a multi-scale and multi-orientation wavelet whereas CLEBP extracts features from a macro space - the information quantity space. 2) ScatNet is a macro feature method whilst CLEBP extracts features from a macro-space using miro-LBP feature method

LBP, are combined.

6.1 Introduction

LBP method and ScatNet are two contrary approaches for texture-image features. The former extracts well micro-structure information from texture, has measures for dealing with image rotation, scaling, and illumination changes while losing macro-structure information. CLBP [24] is a LBP variants which aims at solving the information lost of the original LBP [69] by introducing three complementary components, the sign (CLBP_S), magnitude (CLBP_M), and center pixel intensity (CLBP_C). However, the method is still a type of micro-features. This is proved by the fact that it has a high performance on micro structure image dataset like Outex (with the avarage of 98.34% on the three test suites TC10, TC12_t184, and TC12_horizon) but low classification accuracy on the macro-structure KTH-TIPS2b texture dataset (63.20%).

Contrary to the micro-structure descriptors of LBP family, ScatNet [35] is a macro-feature approach which extracts features by concatenating the coefficients of a 3-layer handcrafted convolution network. In fact, the coefficients of ScatNet are formed by the mean values of filter responses. Therefore, as many other filter bank based methods, ScatNet is also a macro-feature method which usually mislays micro-structure information. The evidence for this is that it has decent classification results on macro texture databases such as KTH-TIPS2b (67.26%) but low accurate rate on micro data, with the average of 97.66% on the three test suites TC10, TC12_t184, and TC12_horizon.

With the aim of leveraging both fine details and broader-range multi-path of signals, we propose a novel framework which combines LBP features and those of scattering transform for texture classification. A preprocessing algorithm, the biologically-inspired filtering(BF) [102], as well as an efficient PCA classifier are also used. Our hand-crafted descriptor has the accuracy which is close to those of the CNN state-of-the-art (FV-CNN [179]) on the three out of four well-known texture datasets (UIUC [52], CUReT [47], KTHTIPS2b [173]), while demonstrating its superior on Outex [9]. FV-CNN is the Fisher Vector aggregation of a Convolutional Neural Network (CNN) filter bank.

In this chapter, it is demonstrated that texture classification can be tackled more effectively by employing the combination of the LBP-and-ScatNet-based complementary features. It is based on publications appeared in [180–182].

6.2 LBP-and-ScatNet-based framework

A motivation for our proposed algorithm is that LBP features and those of scattering transform provide complementary information. The CLBP is used to capture small and fine detail information from texture by encoding the pixel-wise structure in a certain radius. It is usually of from one to seven pixel-distances from a cen-

ter. However, it does not take into account the wider range pixel relationship that takes place beyond the coverage of its radii. In this section, we propose a framework which complements the BF+CLBP features with the broader range, multi-scale, multi-path counterparts, the scattering transform of texture image.

6.2.1 BF+CLBP

Biologically inspired filtering (BF) presented in section 5.2.2 is a preprocessing technique which uses Difference of Gaussians (DoG) filter to imitate the function of retina. DoG is often used as an approximation of a Laplacian of Gaussian (LoG) filter due to its low computational cost. It calculates the second spatial derivative of an image. In areas where images have constant intensity, the filter responses will be zero. Wherever an intensity change occurs, the filter will give a positive response on the darker side and a negative response on the lighter side. In other words, the DoG filter splits the image details alongside two sides of the edges. For a given input image I_{in}, BF of I_{in} outputs two images, let us denote them I_{bf}^{-} and I_{bf}^{+}. Thus, in the feature extraction step with BF used as a preprocessing technique, instead of the input image I_{in}, features are first extracted from the BF output– I_{bf}^{-} and I_{bf}^{+}, and then combined together. The resultant feature vector is considered as the descriptor of I_{in}. For example, with the conventional LBP method, the two histograms of LBP codes estimated from I_{bf}^{-} and I_{bf}^{+} are concatenated and considered as the texture representation of I_{in}.

Concerning LBP, we use the completed model proposed in [24], which gathers three individual operators: CLBP-Sign (CLBP_S) which is equivalent to the conventional LBP, CLBP-Magnitude (CLBP_M) which measures the local variance of magnitude, and CLBP-Center (CLBP_C) which extracts the local central information. When using LBP features, only rotation invariant uniform patterns are considered [2].

We also apply the combination strategies presented in [24]. There are two ways to combine the CLBP_S and CLBP_M codes: by concatenation or jointly. In the first way, the histograms of the CLBP_S and CLBP_M codes are computed separately, and then concatenated together. This CLBP scheme is referred to as "CLBP_S_M". In the second way, a joint 2D histogram of CLBP_S and CLBP_M codes is calculated (denoted as CLBP_S/M).

Finally, the three operators, CLBP_S, CLBP_M and CLBP_C, can be combined in two ways, jointly or hybridly. In the first way, a 3D joint histogram of them is built and denoted by "CLBP_S/M/C". In the second way, a 2D joint histogram, "CLBP_S/C" or "CLBP_M/C" is built first, and then is converted to a 1D histogram, which is then concatenated with CLBP_M or CLBP_S to generate a joint histogram, denoted by "CLBP_M_S/C" or "CLBP_S_M/C".

We therefore apply these schemes on the original texture images and on the preprocessed images which are the outputs of BF.

[2] The rotation invariant uniform patterns are often denoted with the subscript ^{riu_2}, e.g., LBP^{riu_2} or LBC^{riu_2}, but for simplicity they are simply denoted by their names, without ^{riu_2}, e.g., LBP

6.2.2 ScatNet configuration

ScatNet performs scattering transform which captures a wide range of signals for features. The configuration proposed in [35] is used for our framework. That is a three-layer handcrafted convolution network with the first layer implemented two operators consecutively: 1) a 2D-Morlet-wavelet transform followed by 2) a non-linear modulus operator. The latter generates signal magnitudes for the next layers. While the second and third layers are computed by a 3-D wavelet transform of signal magnitudes which have been calculated from the previous layer (Figure 6.1). The 3-D wavelet transform is implemented by a spatial 2D-wavelet transform and a 1D-wavelet transform along the rotation angles. The wavelet transform used here is Morlet wavelet.

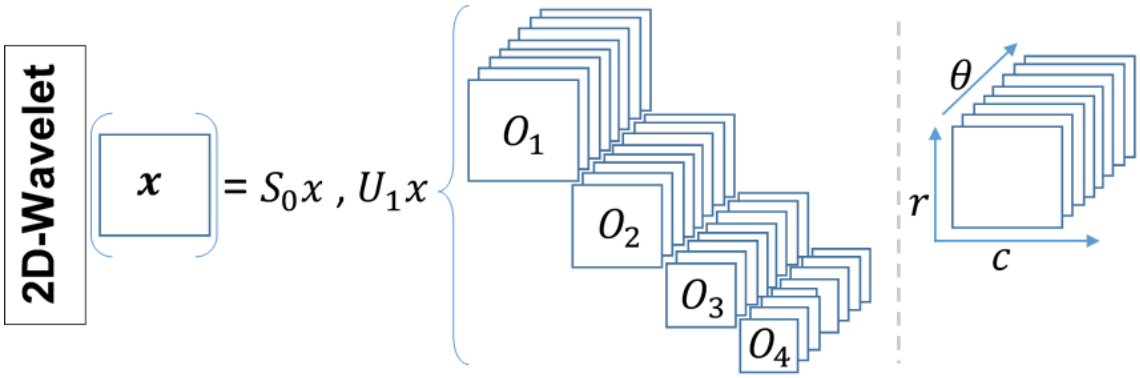


Figure 6.1: A spatial 2D-wavelet on input image x outputs a coefficient S_0x, and U_1x is then grouped into orbits (signals with the same scale and different orientations, left).
3D-wavelet on each orbit (right) is computed by a spatial 2D-wavelet followed by an 1D-wavelet transform along the orientation variables θ.

6.2.3 PCA classifier

A generative classifier called Principal Component Analysis (PCA) [35] was proved to have decent performance for ScatNet. PCA Classifier is described as follows.

Given a test image X, $\widetilde{S}X$ denotes the scattering transform of X and its dilated version D_jX.

$$\widetilde{S}X = \left(\sum_{0 \leq j < H} 1 \right)^{-1} \sum_{0 \leq j < H} \widetilde{S} D_j X. \tag{6.1}$$

The representation of $\widetilde{S}X$ used at test time is therefore a Scattering Transform.

Let $P_{U_c}\widetilde{S}X$ denotes the orthogonal projection of $\widetilde{S}X$ in the scattering space U_c of a given class c. The principal components space U_c is approximately computed

from the singular value decomposition (SVD) of the matrix of centered training sample $\widetilde{S}D_jX_{c,i} - \mu_c$ with all possible samples i dilated by 2^j for a given class c. The PCA classification computes the class $\hat{c}(X)$ base on the minimum distance $\left\|(Id - P_{V_c})(\widetilde{S}X - \mu_c)\right\|$ from $\widetilde{S}X$ to the space $\mu_c + U_c$, Figure 6.2

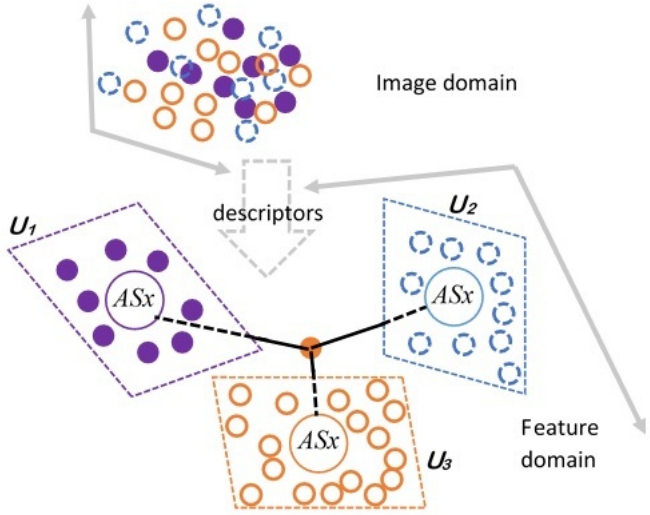


Figure 6.2: PCA-classifier classifies a test image X based on the minimum distance from Scattering Transform $\widetilde{S}X$ to subspace $\mu_c + U_c$.

6.2.4 Framework

The overall idea is illustrated in Figure 6.3. For a given input image x, we first apply BF+CLBP and the Scattering Transform on the image independently, then aggregate their outputs (features) to form final features. BF splits the input image x into I_{bf}^{+} and I_{bf}^{-}, BF+CLBP representation of image x is the combination of CLBP features of I_{bf}^{+} and I_{bf}^{-}, namely BF+CLBP features. Similarly, the ScatNet features are gained by an average pooling of the its coefficients (S_0x, S_1x, S_2x). Finally, an aggregation of those forms our integrated features.

6.3 Experimental validation

This section evaluates the proposed method for classifying texture data. First, parameter settings and datasets are presented. Second, we evaluate the results and compare with the state-of-the-arts. Finally, we analysis the proposed framework and its complexity.

6.3.1 Experimental settings

We analyze the effectiveness of our method by doing experiments on four popular texture databases, and their testing protocols are strictly followed. The PCA

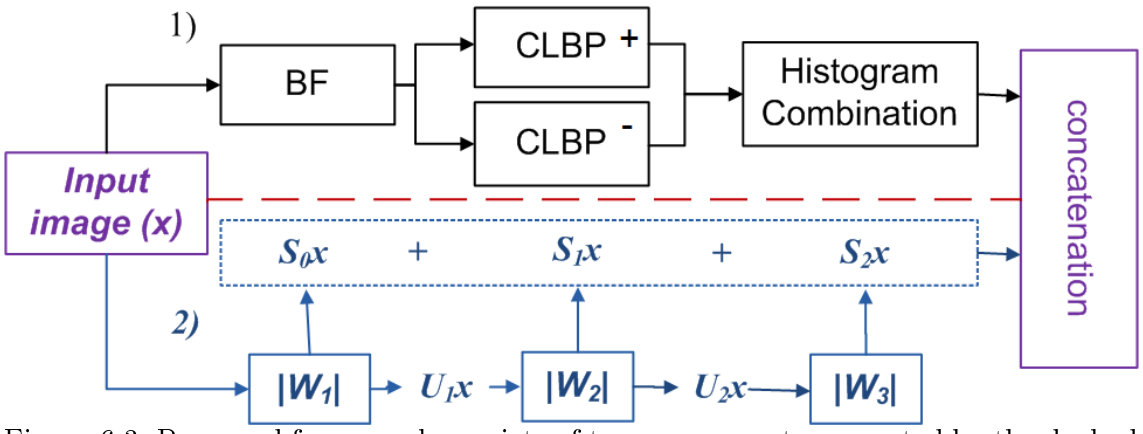


Figure 6.3: Proposed framework consists of two components separated by the dashed line:
1) BF takes one image as input, and generates two outputs I_{bf}^{+} and I_{bf}^{-} which are further used as CLBP's inputs, namely BF+CLBP component.
2) ScatNet component: scattering representation is computed by a cascade of wavelet-modulus operators $|W_m|$. Every $|W_m|$ has one input and two outputs which are the scattering coefficients $S_m x$ and the next layer wavelet modulus coefficients $U_{m+1}x$. The latter is used for further transformation.
Final features are the concatenation of those generated by the components.

classifier [35] instead of a nearest neighbor classifier is used to produce texture classification results because PCA classifier was proved in [35] as a high performance classifier for ScatNet. Also, we want to follow a consistent classification procedure with ScatNet [35] for a fairly comparison. It is worth noting that in the previous chapter (Chapter 5), a nearest neighbor classifier (NNC) is used because we want to prove that there is an enhancement of the proposed method in comparison with other LBP variants which used NNC. In the experiments, we used source code of CLBP, BF, ScatNet, VLFeat library and MatConvNet shared by Guo *et al.*, [24], Vu *et al.* [102], Mallat's group [35], and Vedaldi *et al.* [183, 184] respectively to generate texture classification results.

Databases

We analyze the effectiveness of our method by doing experiments on four popular texture databases with details presented in Section 2.2, and their testing protocols are strictly followed (as summarized in Table 6.1).

The **Outex** [9] database contains texture images captured from a wide variety of real materials. We consider the two commonly used test suites, Outex_TC_00010 (TC10) and Outex_TC_00012 (TC12), containing 24 classes of textures which were collected under three different illuminations ("horizon", "inca", and "t184") and nine different rotation angles (0°, 5°, 10°, 15°, 30°, 45°, 60°, 75° and 90°). There are 20 non-overlapping 128×128 texture samples for each class under each situation.

Table 6.1: Summary of Datasets for the experiments

Dataset	Images	Image size	Classes	Splits
CURet	5612	200×200	61	100
UIUC	1000	640×480	25	100
Outex TC10	4320	128×128	24	pre-defined
Outex TC12	9120	128×128	24	pre-defined
KTH-TIPS2b	4752	200×200	11	pre-defined

The 24×20 samples with illumination condition "inca" and rotation angle 0° were adopted as the training data.

The **CUReT** [47] database contains 61 texture classes, each having 205 images acquired at different viewpoints and illumination orientations. There are 118 images shot from a viewing angle of less than 60°. From these 118 images, as in [24, 174], we selected 92 images, from which a sufficiently large region could be cropped (200×200) across all texture classes. All the cropped regions are converted to grey scale (examples are shown in Figure 2.4).

The **UIUC** [52] texture database includes 25 classes with 40 images in each class. The resolution of each image is 640×480. The database contains materials imaged under significant viewpoint variations (examples are shown in Figure 2.8).

The material databases **KTHTIPS2b** [173], with 3 viewing angles, 4 illuminants, and 9 different scales, produce 432 images per class, with the image size of 200×200 and 11 classes in total. Regarding the KTHTIPS2b databases, we follow the common testing and training protocols. Only unseen data is used for testing, with three out of four samples used for training and the remaining for testing.

6.3.2 BF+CLBP settings

BF parameter

Parameters of the BF filter include the two standard deviations σ_1 and σ_2 defining the low and high cutoff frequencies of the band pass *DoG* filter, and the threshold ϵ. It is worth noting that multiple bands can be used. For example, when using three different BF filters with different cutoff frequencies, one can obtain six different "maps" where the feature extraction is applied. However, on the one hand, the multiple bands increase both the feature extraction and classification time, and on the other hand, this section focuses rather on the descriptors, we do not report the result of multiple bands. A critical constraint is $\sigma_1 < \sigma_2$ (recall that these correspond respectively to the parameters of the blurring filters being performed in photoreceptors and horizontal cells). By varying σ_1, σ_2, and ϵ ($\sigma_1 \in \{0.5, 0.75, 1, 1.25, 1.5\}$,$\sigma_2 \in \{2, 3, 4, 5, 6\}$, $\epsilon \in \{0.05, 0.1, 0.15, 0.2, 0.25, 0.3\}$),

we used the BF parameters in this chapter as follows: $\sigma_1 = 1$, $\sigma_2 = 5$, and $\epsilon = 0.15$.

CLBP

We use the completed model proposed in [24] for extracting LBP features. In which three individual LBP operators applied on the BF maps of an input image: 1) CLBP-Sign (CLBP_S) which is equivalent to the conventional LBP, 2) CLBP-Magnitude (CLBP_M) which measures the local variance of magnitude, and 3) CLBP-Center (CLBP_C) which extracts the local central information. When using LBP features, only rotation invariant uniform patterns are considered [3].

We also apply the combination strategies presented in [24], the CLBP_S-and-CLBP_M-code concatenation or jointly. In the former, the histograms of the CLBP_S and CLBP_M codes are computed separately, and then concatenated. This CLBP scheme is referred to as "CLBP_S_M". In the latter, a joint 2D histogram of CLBP_S and CLBP_M codes is calculated (denoted as CLBP_S/M).

At the end, three operators, CLBP_S, CLBP_M and CLBP_C, can be combined in a joint or hybrid manner. In the first way, a 3D joint histogram of them is built and denoted by "CLBP_S/M/C". In the second way, a 2D joint histogram, "CLBP_S/C" or "CLBP_M/C" is built first, and then is converted to a 1D histogram, which is then concatenated with CLBP_M or CLBP_S to generate a joint histogram, denoted by "CLBP_M_S/C" or "CLBP_S_M/C".

In addition, 3-scale CLBP is also utilized in our experiments. A popular scheme of radii and number of neighbors (R,P) are applied, they are (1,8), (2,16), and (3,24).

6.3.3 ScatNet settings

ScatNet parameters are selected such that scaling number is $J = 3$ for datasets with resolution of images at 150×150 or below, $J = 4$ for image size 300×300 or smaller, and $J = 5$ otherwise. Here, the principle is that the smaller image size, the smaller scaling number chosen as recommended in [35]. Orientations of filter bank is set to 8 ($L = 8$), the number of ScatNet layers is set to 3 ($M = 3$). Since if the layer number exceeds this threshold, both feature dimension and feature extraction time increase with minor improvement in classification accuracy.

The following sections will discuss about classification results of our framework and compare them with those of recent state-of-the-arts.

[3] The rotation invariant uniform patterns are often denoted with the subscript ^{riu_2}, e.g., LBP^{riu_2} or LBC^{riu_2}, but for simplicity they are simply denoted by their names, without ^{riu_2}, e.g., LBP

6.3.4 Results on the Outex dataset

In our framework, there are actually two types of complementary features, the micro-structure (LBP) and the macro counterpart, ScatNet. Therefore, the first step is extracting 3-scale LBP features with the scheme of popular radii $R = 1, 2, 3$, and number of neighbors $P = 8, 16, 24$, the CLBP variant is used in this case. In the second step, coefficients of ScatNet are aggregated, we use the original aggregation method proposed in [35], the mean values of coefficients. The final features are the concatenation of those two descriptor types, the micro structure features are fused with the macro counterpart results in a novel robust texture representation approach for variety of type of texture. This property is verified on Outex and three other datasets.

There are actually two experiments on each dataset. 1) ScatNet is combined with CLBP features. 2)ScatNet combines with BF+CLBP features.

Given an image, the BF filtering is first applied and the two maps are obtained (see I_{bf}^{+} and I_{bf}^{-} in Figure 5.3). The CLBP_SMC features are extracted on both I_{bf}^{+} and I_{bf}^{-}. All resulting descriptors are then concatenated in order to obtain the final descriptor representing the considered texture image. Recall that, in the three tests of this database, the training and testing images are priorly split by its predefined evaluation protocol.

Table 6.2 compares the classification results obtained by our proposed framework in this chapter with those of our work presented in the previous chapter, and other state-of-the-arts. PCA classifier [35] is used for our proposed methods to prove for the improvement of the proposed method in comparison with the ones it derived from. The features include single scale CLEBP, multi-scale CLBP, ScatNet, and the proposed framework are classified by PCA classifier in this experiment.

It is worth noting that our proposed method in Chapter 5, the CLEBP has the highest performance on this dataset, with the average accuracy at 99.61 %, while our work presented in this chapter, the micro-macro complementary feature, is ranked second. However, our framework in this chapter outweighs the CLEBP in the more challenging datasets with scale variance. The reason can be easily noticed is that the framework has the ability of extracting both micro and macro structure information of images while the CLEBP does not, this can be seen in the next sections.

Here are some most interesting observations:

1) The texture classification accurate rates are at 99.87%, 98.43%, and 99.63% for Outex_TC10, Outex_TC12_00, and Outex_TC12_01 respectively, they are consistently higher than those of the original features. Those classification results of ScatNet are, in turn for the three test suites, 98.39%, 96.23%, and 98.38% while these figure for CLBP and BF+CLBP are respectively 99.20%, 96.20%, 97.25% and 99.50%, 97.13%,98.40%.

2) Our results are much better than those of FV-SIFT+ FC+FV-VD [179], which can be considered as the frontier on many datasets other than Outex, the data suite is mainly used to test the rotation invariance of features. One can observe that the two rotation invariance features, ScatNet and CLBP, are combined together in our novel framework, this results in the features which are also invariant to rotation yet more robust. In fact, FV-SIFT+ FC+FV-VD [179] is a very powerful framework which is formed by combining features from a 19-layer pre-trained CNN and SIFT features. The final Fisher Vector aggregated features of this framework show a competitive classification results on many texture datasets except the Outex. This is caused by the data dependence of the pre-trained CNN. On contrary, our handcrafted method in this chapter, without any expensive learning stage, can deal well with such a variety type of texture. Details can be seen in Table 6.2.

3) MRELBP[34] is a LBP variant which gains significant performance with multiscale version being up to nine scales. It should be noticed that we choose CLBP instead of others to combine with ScatNet because we want to prove that a basic LBP variant, which is closed to the original rather than an advanced version, and ScatNet can be complementary.

6.3.5 Results on the CUReT database

In the experiments on the CUReT database, as in [24], to get statistically significant experimental results, N training images were randomly chosen from each class while the remaining $92 - N$ images per class were used as the test set. We pick up the high performance results, with $N = 46$, for a solid report in this chapter.

Similar to experiments on the Outex database, we compare the framework classification results with our proposed method (single scale CLEBP with PCA classifier) presented in the previous chapter, and other state-of-the-art methods. We include different features: VZ-MR8, LBP and its variants, such as LBP/VAR, DLBP (Dominant LBP), multi-scale CLBP_SMC, as well as the two very recent algorithms, MRELBP [34]. The highest performance of all reported results chosen for comparison, some benchmarks used SVM classifier. The average classification rates on those databases over a hundred random splits with different parameters are reported in Table 6.3, from which the efficiency of our approach obtained by its complementary components can be noticed. When combining 3-scale CLBP and ScatNet, the obtained results are better than all other LBP variants, and ScatNet. Frankly, features extracted from a 9-layer pre-trained CNN combined with those of SIFT, the FV-SIFT+ FC+FV-VD[179], gained the best performance on this dataset. However, it should be recalled that our handcrafted method has its components which consist of a compact feature type (LBP), and a three-layer convolution network without learning stage. Even though, we have a significant classification results on Outex, whereas the FV-SIFT+ FC+FV-VD[179] dose not.

Our proposed framework has a slight improvement comparing to the method it

Table 6.2: Classification accuracy (%) comparisons with state-of-the-art on the Outex database. (⋄): Our method in Chapter 5; (⋄⋄): Our method in this chapter.

Method	TC10	TC12		
		t184	horizon	mean
CLEBP (⋄)	99.87	99.38	99.59	99.61
CLBP_S/M+ScatNet (⋄⋄)	99.40	98.45	98.80	98.88
CLBP_S/M/C+ScatNet (⋄⋄)	99.51	98.43	98.94	98.96
BF+CLBP_S/M+ScatNet (⋄⋄)	99.82	98.22	99.40	99.15
BF+CLBP_S/M/C+ScatNet (⋄⋄)	99.87	98.66	99.63	99.39
BF+CLBP_S/M	99.60	97.29	97.99	98.29
BF+CLBP_S/M/C	99.50	97.13	98.40	98.34
CLBP_S/M	99.30	96.40	97.05	97.58
CLBP_S/M/C	99.20	96.20	97.25	97.55
ScatNet[35]	98.39	96.23	98.38	97.67
MRELBP[34]	99.38	99.17	99.51	99.35
NTLBP[131]	99.24	96.18	94.28	96.57
DLBP[71]	99.10	93.20	90.04	94.11
COV-LBPD[185]	98.78	95.72	97.62	97.37
LTP[70]	98.54	92.59	89.17	93.43
LBP [69]	97.70	87.30	86.40	90.47
MSJLBP[136]	96.67	95.21	95.74	95.87
PRICoLBP[135]	94.48	92.57	92.50	93.18
VZ-MR8[176]	93.59	92.55	92.82	92.99
NRLBP[186]	93.44	86.13	87.38	88.98
VZ-Path [174]	92.00	91.41	92.06	91.82
MBP[187]	89.92	95.18	95.55	93.55
FV-SIFT+ FC+FV-VD[179]	80.00	82.16	82.44	81.53

derived from, the CLBP and ScatNet. As illustrated in Table 6.3, the classification rate of our method closely reaches the state-of-the-art on this dataset. Once again, the small fined structure features (CLBP) can compensate for scattering transform features for being robust to the combinations of viewing and illumination direction changes of texture in the CUReT dataset.

It is worth noticing that our first proposed texture representation method, the single scale CLEBP presented in Chapter 5, has very competitive results on Outex but it does not on CURet. This is because the structure information it captures from images is not enough diversity as our second proposed method does, the micro-macro complementary feature approach.

Table 6.3: Classification accuracy comparisons with state-of-the-art on the CUReT database. ($\diamond$): Our method in Chapter 5; ($\diamond\diamond$): Our method in this chapter.

Method	Accuracy(%)
CLEBP ($\diamond$)	98.92
CLBP_S/M+ScatNet ($\diamond\diamond$)	99.53
CLBP_S/M/C+ScatNet ($\diamond\diamond$)	99.50
BF+CLBP_S/M+ScatNet ($\diamond\diamond$)	**99.74**
BF+CLBP_S/M/C+ScatNet ($\diamond\diamond$)	99.51
CLBP_S/M	98.60
CLBP_S/M/C	98.56
BF+CLBP_S/M	98.65
BF+CLBP_S/M/C	98.57
ScatNet[35]	99.10
FV-SIFT+ FC+FV-VD[179]	99.70
Broadhurst[188]	99.22
MRELBP [34]	99.02
VZ-MR8[176]	98.61
Multi-scale BIF[189]	98.60
Hayman *et al.*[190]	98.46
VZ-joint[174]	97.66
Zhang *et al.*[191]	95.30
DLBP[71]	92.77

6.3.6 Results on the UIUC database

This section discusses in more detail the classification results obtained on the UIUC dataset. Experiment protocol on the UIUC database is similar to that on CUReT, to eliminate the dependence of the results on the particular training images used, N training images were randomly chosen from each class while the remaining $40 - N$ images per class were used as test set. There are two experiments actually drawn on this dataset.

1) CLBP+ScatNet: We get around 2% classification enhancement on this dataset (from 95.90% to 97.50%) comparing to its original version.

2) BF+CLBP+ScatNet has got similar improvement in classification accuracy, increasing from 95.20% to 98.69%. When BF preprocessing techniques added to CLBP, the accurate rate rises roundly 1%.

From above analyses, one can conclude that local CLBP/BF+CLBP features combined with broader range features (ScatNet) will improve the overall performance on UIUC dataset. This is because these two kinds of features are complementary in such a way that they can be robust to the changes of viewing angle, scale, and illumination conditions.

Results in Table 6.4 shows that our accurate rate on this dataset is similar to that of works in [191], roundly 1% lower than the recent frontier FV-SIFT+ FC+FV-VD [179] of this dataset while beating all others.

Table 6.4: Classification accuracy comparisons with state-of-the-art on the UIUC database. ($\diamond$): Our method in Chapter 5; ($\diamond\diamond$): Our method in this chapter.

Method	Accuracy(%)
CLEBP ($\diamond$)	95.81
CLBP_S/M+ScatNet ($\diamond\diamond$)	97.20
CLBP_S/M/C+ScatNet ($\diamond\diamond$)	97.50
BF+CLBP_S/M+ScatNet ($\diamond\diamond$)	98.63
BF+CLBP_S/M/C+ScatNet ($\diamond\diamond$)	**98.69**
CLBP_S/M	95.90
CLBP_S/M/C	96.02
BF+CLBP_S/M	96.48
BF+CLBP_S/M/C	96.28
ScatNet[35]	95.20
FV-SIFT+ FC+FV-VD[179]	99.90
Zhang *et al.*[191]	98.70
WMFS[191]	98.60
VZ-joint[174]	97.83
OTF[192]	97.40
MFS[193]	92.74
Hayman *et al.*[190]	92.00

6.3.7 Results on the KTH-TIPS-2b

For the experiments on KTHTIPS2b, the training and testing scheme used in [173] are followed. We conduct experiments by training on one, two, or three samples; testing is always conducted only on un-seen ones. For example, there are six experiments where two samples are used as training and the two remaining samples are used as testing, and the classification rate reported is the average of those

results. In this challenging database, our the proposed framework, the LBP-and-ScatNet-based complementary approach demonstrates its superior strength to other handcrafted features including our efficient LBP variant, the CLEBP presented in Chapter 5. Table 6.5 details our results obtained on this challenging dataset with different combination strategies. It also compares the framework classification accurate gain with other recent well-known state-of-the-art methods. As can be seen in the table, our approach considerably outperforms almost all competing algorithms, except the 19-layer-CNN and SIFT hybrid method with Fisher Vector aggregation, the FV-SIFT+ FC+FV-VD[179]. The best results of the highest performance (MRELBP[34]) among handcrafted descriptors is 77.91% while our result is 78.09 %. This performance is much better than our previous proposed method, the CLEBP with PCA classifier whose result on this dataset is at 70.50 %.

As can be seen from Table 6.5, our proposed framework gets a significant improvement over the original features it inherits from, the enhancement in classification accuracy is approximately 6% for CLBP+ScatNet, 10% for BF+CLBP+ScatNet, and up to 14% when comparing to CLBP or ScatNet features alone. Also, our classification result on this database is ranked closely to FV-SIFT+ FC+FV-VD [179]. As discussed earlier, the method of our second contribution can demonstrate its superior strength to our first contribution because it can leverage the micro-structure information from LBP features and the macro-structure information from ScatNet descriptors. Surprisingly, the two type of feature are complementary in our framework, this results in a discriminative handcrafted texture representation.

6.3.8 Framework experimental analysis and complexity

In this section, we first show the complementary strength of CLBP and ScatNet in the proposed framework, features generated from the framework are more tolerant to the variations in scale, pose and illumination with the KTH-TIPS-2b dataset when CLBP and ScatNet combined. Then, we represent the complexity of our method (Table 6.9). Finally, we discuss about the use of feature dimensionality reduction and its corresponding accuracy.

Tables 6.6, 6.7, and 6.8 present the confusion matrices of classification of one split on the KTH-TIPS-2b dataset with BF+CLBP, ScatNet, and the combined features respectively. The results show that our framework leverages the complementary properties of CLBP (micro-structure information features) and ScatNet (macro-features) to reduce the misclassification cases, confusion matrix of the framework (Table 6.8) shows a smaller miss-class numbers than those of the other two (Tables 6.6, 6.7). Figure 6.4 illustrates the cases of class 4 (cork) and 7 (lettuce Leaf) are classified incorrectly by both CLBP and ScatNet. However, the proposed framework perfectly classify images in these classes. This shows that CLBP and ScatNet can complement each other. Figure 6.4 illustrates misclassification cases of the original features while the proposed method classify these images without error. However, BF+CLBP mis-classifies two left images: **cork** (class 4) is classified incorrectly into

Table 6.5: Classification accuracy comparisons with state-of-the-art on the KTH-TIPS-2b database. (⋄): Our proposed method in Chapter 5; (⋄⋄): Our proposed method in this chapter

Method	Accuracy(%)
CLEBP (⋄)	70.50
CLBP_S/M+ScatNet (⋄⋄)	69.15
CLBP_S/M/C+ScatNet (⋄⋄)	68.80
BF+CLBP_S/M+ScatNet (⋄⋄)	**78.09**
BF+CLBP_S/M/C+ScatNet (⋄⋄)	77.71
CLBP_S/M	63.30
CLBP_S/M/C	63.20
BF+CLBP_S/M	67.68
BF+CLBP_S/M/C	68.31
ScatNet[35]	67.26
FV-SIFT+ FC+FV-VD[179]	81.50
MRELBP[34]	77.91
MSJLBP[136]	65.51
ELBP[74]	64.84
MWLD[151]	64.70
COV-LBPD[185]	63.47
LTP[70]	62.12
PRICoLBP[135]	61.17
LBP [69]	60.35
MBP[187]	60.29
VZ-Path [174]	60.70
NTLBP[131]	58.78
NRLBP[186]	57.00
VZ-MR8[176]	55.7

cracker (class 6), brown bread (class 2), corduroy (class 3), and cotton (class 5). While **lettuce leaf** (class 7) is classified incorrectly into aluminium foil (class 1). ScatNet mis-classifies 2 right images: **cork** (class 4) is misclassified into aluminium foil (class 1), brown bread (class 2), corduroy (class 3), lettuce leaf (class 7), and white bread (class 9), **lettuce leaf** (class 7) is misclassified into aluminium foil (class 1).

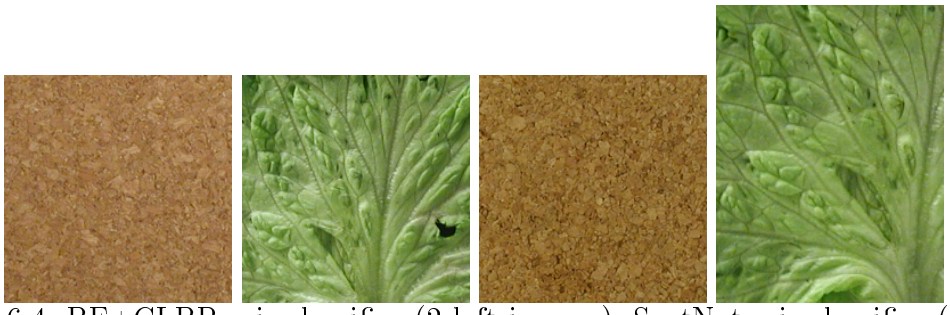


Figure 6.4: BF+CLBP mis-classifies (2 left images), ScatNet mis-classifies (2 right images), the proposed method classifies these images without error.

Among those methods mentioned in this chapter, network-based approaches have high classification accuracy and obviously high complexity in comparison with oth-

Table 6.6: Confusion Matrix BF+CLBP on the KTH-TIPS-2b database.

108	0	0	0	0	0	0	0	0	0	0
0	61	0	0	0	7	0	0	40	0	0
0	1	95	0	7	0	0	2	0	1	2
0	1	1	103	1	2	0	0	0	0	0
0	0	0	0	39	2	0	54	5	6	2
0	8	24	7	0	56	2	0	11	0	0
3	0	3	0	0	1	96	0	0	5	0
0	0	0	0	5	0	0	96	0	0	7
0	8	1	0	0	6	0	0	92	1	0
0	0	1	0	0	0	0	0	0	107	0
0	0	11	0	4	0	14	7	0	0	72

Table 6.7: Confusion Matrix Scatnet on the KTH-TIPS-2b database.

108	0	0	0	0	0	0	0	0	0	0
15	44	1	2	0	8	14	0	21	2	1
1	0	101	0	1	0	0	3	0	2	0
5	12	1	77	0	0	2	0	11	0	0
0	0	2	0	64	0	4	30	0	7	1
3	11	7	12	0	38	19	0	9	4	5
19	0	0	0	0	0	89	0	0	0	0
0	1	0	0	6	1	0	99	0	0	1
3	13	2	10	0	2	12	0	62	4	0
11	0	0	0	0	0	1	0	0	96	0
11	3	25	0	0	0	10	0	0	3	56

Table 6.8: Confusion Matrix of the proposed method on the KTH-TIPS-2b database.

108	0	0	0	0	0	0	0	0	0	0
0	97	0	0	0	8	0	0	3	0	0
0	0	98	0	8	0	0	1	0	1	0
0	0	0	108	0	0	0	0	0	0	0
0	0	1	0	65	0	0	30	0	7	1
0	17	12	8	0	64	1	0	3	0	3
0	0	0	0	0	0	108	0	0	0	0
0	0	0	0	2	0	0	103	0	0	3
0	0	1	0	0	0	0	0	107	0	0
0	0	1	0	0	0	0	0	0	107	0
0	0	0	0	0	0	19	5	0	0	84

ers. We will present the complexity of our work, and compare it with the state-of-the-art CNN, the FV-SIFT+ FC+FV-VD[179] (Table 6.9). Feature extraction time (Fea. Ex. Time) of methods in second, is calculated on the same laptop with Intel(R) Core(TM) i7-4710MQ CPU @ 2.50GHz, 2501 Mhz, 4 Core(s), 8 Logical Processor(s) and 8Gb of RAM. We measure the feature extraction time by the average computation time of 500 images from the KTH-TIPS-2b dataset.

Our framework feature extraction time depends on ScatNet parameters (scales), the smaller scale, the faster feature extraction. In general, our approach takes a little longer time to extract features in comparison with FV-SIFT+ FC+FV-VD[179]. However, our method has the feature dimension which is much lower, this leads to a lower classification time. We also examined the performance of BF + CLBP, followed

Table 6.9: Framework complexity and comparison, feature extraction time in seconds (Feat. Ex. Time(seconds)), feature dimension (Feat. Dimension).

Method	Feat. Ex. Time(second)	Feat. Dimension
Ours	0.7073	4,608
FV-SIFT+ FC+FV-VD[179]	0.5431	65,536

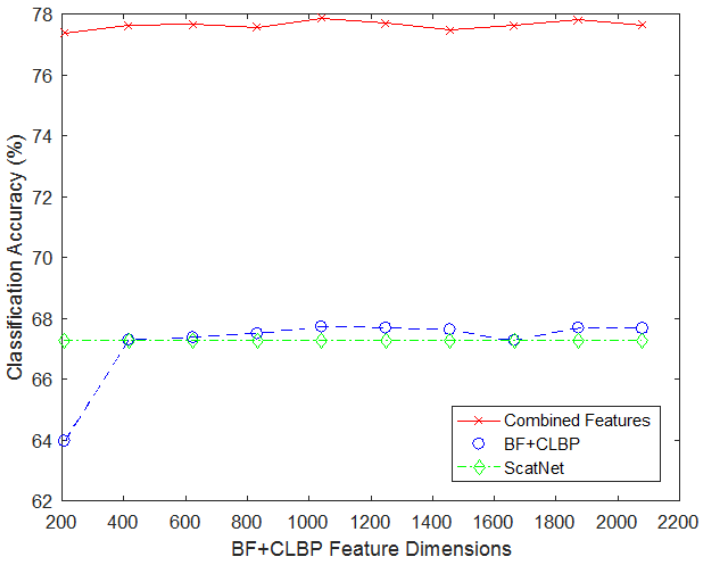


Figure 6.5: Classification accuracy on the KTH-TIPS-2b database where PCA dimensionality reduction is done upon the BF+CLBP-feature-vectors before concatenating with those of ScatNet.

by the PCA dimensionality reduction technique. Figure 6.5 presents the classification accuracy on KTH-TIPS-2b dataset with such features. There is a slight decrease in classification accuracy (from 68.38% to 67.28%) when dimension of BF+CLBP feature vectors are reduced from 3888 to 416. However, the accurate rates of the combined method are stable around 77.63% with feature vector dimension ranging from 416 to 2288 (the feature vector dimension without dimension reduction is 4096), the accuracy peaks at 77.87% (the accurate rate without dimension reduction

is 78.09%) with the feature vector dimension of 1248. Therefore, applying feature vector dimension reduction of BF+CLBP features before concatenating with those of ScatNet can be considered as a trade-off between classification accuracy and feature dimension.

6.4 Conclusions

In this chapter, we have proposed a novel framework which generates handcrafted features for texture classification. It takes full advantages of the BF preprocessing technique, the local LBP features, and the global ones extracted from ScatNet. Experiments show our proposed method enhances distinctiveness of texture while preserving the robustness to variations in illumination, rotation, and scale. Overall, ScatNet and LBP are not concurrent but complementary while the preprocessing technique makes the LBP descriptors more robust. Our modest improvements in terms of classification accuracy have not reached the recent frontiers but are in the high ranking on the experimental databases, to the best of our knowledge. Future study can be drawn on the same domain with scale variation tolerance by using multi-scale training techniques. Next chapter we will introduce a novel ScatNet variant which enhances the strength of ScatNet for texture representation, namely a Normalized-Convolution Network.

Chapter 7

Normalized-Convolution Network: A More Efficient Net-based Method

So far, we have proposed two different methods for texture representation. The first contribution focused on developing a LBP variant which extracts features based on complementary signals from input images, they are micro-structure information from LBP operators and information quantity computed using entropy. On contrary, our second contribution proposes a compound framework which leverages the strength of two types of features which are not concurrent yet complementary, the micro-features extracted from LBP and a macro counterparts derived from ScatNet.

In this chapter, we introduce a Handcrafted Normalized-Convolution Network (NmzNet) for efficient texture classification. NmzNet is implemented by a three-layer normalized convolution network, which computes successively normalized convolution with a predefined filter bank (Gabor filter bank) and modulus non-linearities. Coefficients from different layers are aggregated by Fisher Vector aggregation to form the final discriminative features. The results of experimental evaluation on three texture datasets UIUC, KTH-TIPS-2a, and KTH-TIPS-2b indicate that our proposed approach achieves the good classification rate compared with other handcrafted methods. The results additionally indicate that only a marginal difference exists between the best classification rate of recent frontiers CNN and that of the proposed method on the experimented datasets.

7.1 Introduction

Convolution is an essential operator which is used in almost every image processing technique, and texture analysis is not an exception. Let us consider texture classification for an example, filter bank based methods have played a vital role

with several successful proposed works where the fundamental operator of those is the convolution. Bovik *et al.* applied the Gabor filters to compute the average filter responses for features [77]. Mallat proposed the multi-resolution wavelet decomposition method [58], which generates coefficients from the high-low (HL), low-high (LH), and low-low (LL) channels for subsequent classification tasks. Porter *et al.* [78] removed the high-high (HH) wavelet channels and combined the LH and HL wavelet channels to obtain rotation invariance wavelet features. Haley *et al.* [79] calculated isotropic rotation invariance features from Gabor filter responses. More recent, scattering transform is considered as a high performance approach based on cascading wavelet transform layers [35] compared to previous wavelet-based methods.

There is another operator which can be computed using standard convolution called normalized-convolution [21]. It has the potentiality of a decent fundamental operator for texture descriptors because it is based on the concept of the signal/certainty - philosophy to produce a description of the neighborhood which is optimal in the mean square sense. We therefore propose to use normalized-convolution in the scattering network model to substitute for standard convolution for texture classification. The idea was first introduced in [194].

This chapter is organized as follows: we start with a review of the normalized convolution in Section 7.2.1, Fisher vector aggregation presented in Section 7.2.2, followed by details of the proposed Cormalized-Convolution Network (Section 7.2.3). In Section 7.3, we verify the our approach with experiments on popular texture datasets and comparisons with various state-of-the-art texture classification techniques. Section 7.4 provides concluding remarks and possible extensions of the proposed method.

7.2 Normalized-convolution network

In this section, we propose a handcrafted Normalized-Convolution Network with its derivation described in 7.2.1 for efficient texture classification.

7.2.1 Normalized convolution

Normalized convolution [21] is a method for performing general convolution operations on data of signal/certainty type. More specifically, it takes into account the uncertainties in signal values and allows spatial localization of possibly unlimited analysis functions. The conceptual basis for the method is the signal/certainty philosophy, i.e. separating the values of a signal from the certainty of the measurements. The separation of both data and operator into a signal part and a certainty part. Missing data is simply handled by setting the certainty to zero. In the case of uncertain data, an estimate of the certainty must accompany the data. Localization or "windowing" of operators is done using an applicability function, the operator equivalent to certainty, not by changing the actual operator coefficients. Spatially

or temporally limited operators are handled by setting the applicability function to zero outside the window.

Formally, Knutsson *et al.* [21] define normalized convolution in form of standard convolution as follows,

$$U(\xi) = \sum_x a(x)B(x) \odot c(\xi - x)T(\xi - x) \tag{7.1}$$

where
ξ is the global spatial coordinate.
x is the local spatial coordinate.
$T(\xi)$ is a tensor corresponding to the input signal.
$c(\xi)$ is a function which represents the certainty of $T(\xi)$.
$B(x)$ is a tensor corresponding to the operator filter basis.
$a(x)$ is the operator equivalent to certainty, a function which represents the applicability of $B(x)$.

$\odot$ denotes multi-linear operations (in standard convolution this operation is scalar multiplication). When the basic operation is understood explicitly indicating the dependence on the global spatial coordinates ξ, the local spatial coordinate x plays no role. Then, the expression (7.1) can be written as

$$U = \{aB\widehat{\odot}cT\} \tag{7.2}$$

where the "hat" over the multilinear operation symbol acts as a marker of the operation involved in the convolution (it is useful when more than one operation symbol appears within the brackets).

Knutsson *et al.* also draw another definition of normalized convolution by the means of general convolution operations on data of signal/certainty type. Normalized convolution of aB and cT can be defined and denoted by:

$$U_N = \{aB\widehat{\odot}cT\}_N = N^{-1}D \tag{7.3}$$

where: $D = \{aB\widehat{\odot}cT\}$
$N = aB \odot B^{*}\widehat{.}c$
The star, $*$, is the complex conjugate operator.

The concept of normalized convolution has a theory behind, the least squares problem.

To see this, for a given neighborhood, t, in a set of basis functions given by a matrix B and the coefficients u. The approximation t' of t is the projection of u on B, (standard matrix and vector notation are used).

$$t' = Bu \tag{7.4}$$

It is proved that for a given set of basis functions, B , the least square error, $\|t' - t\|$, is minimized by choosing u to be:

$$u = \left[B^T B\right]^{-1} B^T t \tag{7.5}$$

With the introduction of a diagonal matrix, W, a weighted least square problem can be solved by

$$Wt' = WBu \tag{7.6}$$

Then the minimum of $\|W(t' - t)\|$ is obtained by choosing coefficients u to be:

$$u = \left[(WB)^T WB\right]^{-1} (WB)^T Wt \tag{7.7}$$

which can be rewritten and split into two parts, N^{-1} and D.

$$u = \underbrace{\left[B^T W^2 B\right]^{-1}}_{N} \underbrace{B^T W^2 t}_{D} \tag{7.8}$$

It can be shown that N and D are identical to the corresponding quantities used in normalized convolution, equation (7.3). In normalized convolution, the diagonal weight matrix is, for a neighbourhood centered on ξ_0, given by:

$$W_{ii}^2 (\xi_0) = a(x_i)\, c(\xi_0 - x_i) \tag{7.9}$$

Therefore, normalized convolution can be considered as a method used to obtain a local weighted least square error representation of the input signal. The input signal is described in terms of the basis function set, B, while the weights are adaptive and given by the data certainties and the operator applicability. One of the most striking applications of normalized convolution might be the interpolation of Lena image (Figure 7.2) using applicability function illustrated in Figure 7.1.

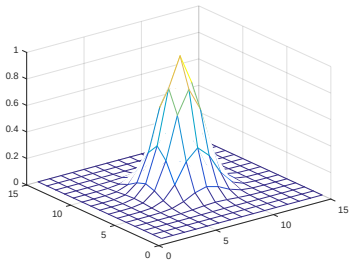


Figure 7.1: An applicability function.

Figure 7.2: Top left: The famous Lena-image, top right the image has been degraded to a grey-level test image only containing 20% of the original information. Bottom left: Interpolation using standard convolution with a normalized smoothing filter (see Figure 7.1). Bottom right: Interpolation using normalized convolution with the same filter as applicability function.

7.2.1.1 Example

To give a small example, assume that we have a two-dimensional signal T, sampled at integer points, with an accompanying certainty field c, as defined below.

$$T = \begin{matrix} 9 & 6 & 9 & 1 & 7 \\ 10 & 10 & 2 & 9 & 2 \\ 2 & 10 & 5 & 10 & 8 \\ 10 & 2 & 10 & 7 & 1 \\ 7 & 10 & 8 & 8 & 3 \\ 1 & 10 & 10 & 8 & 1 \\ 3 & 5 & 7 & 4 & 1 \end{matrix} \qquad c = \begin{matrix} 0 & 2 & 2 & 2 & 2 \\ 2 & 1 & 1 & 2 & 2 \\ 2 & 1 & 1 & 2 & 1 \\ 2 & 2 & 2 & 2 & 1 \\ 1 & 0 & 2 & 2 & 2 \\ 1 & 1 & 2 & 1 & 0 \\ 2 & 2 & 2 & 1 & 0 \end{matrix}$$

Let the local signal model be given by a polynomial basis, $\{1, x, y, x^2, xy, y^2\}$ (it is understood that the x variable increases from the left to the right, while the y variable increases going downwards) and an applicability of the form:

$$a = \begin{matrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{matrix}$$

The applicability fixes the size of the neighborhood, in this case 3×3 pixels, and gives a localization of the unlimited polynomial basis functions. Expressed as matrices,

taking the points columnwise, we have

$$B = \begin{pmatrix} 1 & -1 & -1 & 1 & 1 & 1 \\ 1 & -1 & 0 & 1 & 0 & 0 \\ 1 & -1 & 1 & 1 & -1 & 1 \\ 1 & 0 & -1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & -1 & 1 & -1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \quad and \quad a = \begin{pmatrix} 1 \\ 2 \\ 1 \\ 2 \\ 4 \\ 2 \\ 1 \\ 2 \\ 1 \end{pmatrix}$$

Assume that we wish to compute the result of normalized convolution at the marked point in the signal. Then the signal and certainty neighborhoods are represented by

$$t = \begin{pmatrix} 2 \\ 10 \\ 10 \\ 10 \\ 8 \\ 10 \\ 7 \\ 8 \\ 8 \end{pmatrix} \quad and \quad c = \begin{pmatrix} 2 \\ 0 \\ 1 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 1 \end{pmatrix}$$

Applying equation 7.8 with $W^2 = W_a W_c$ we get the result

$$u = (B^T W_a W_c B)^{-1} B^T W_a W_c t = \{8.4275, 1.3913, 0.9855, -2.6739, -1.5362, 1.1449\}$$

with this choice of basis functions, the resulting coefficients hold much information on the the local orientation of the neighborhood. To conclude this example, we reconstruct the projection of the signal, $t^{'} = B \times u$, and reshape it to a 3×3 neighborhood:

2.9855	8.5870	8.8406
4.3623	8.4275	7.1449
8.0290	10.5580	7.7391

To get the result of normalized convolution at all points of the signal, we repeat the above process at each point.

7.2.1.2 Implementation Matters of Normalized Convolution

In fact, Convolution Networks are implemented by convolving a given input image or some cases a feature map (a filter response of previous convolution) with a set of filters. To build Normalized Convolution Network using normalized convolution as a primal operator, an important question need to be answer is that how can

we use normalized convolution to compute filter responses when a bank of filters is provided?

For a given set of convolution kernels, Farnebäck [195] suggested to conjugate and reflect these kernels to obtain the equivalent correlation kernels $\{h_i\}_1^m$, such that

$$u_i(x) = (h_i \star f)(x) \tag{7.10}$$

The idea is to find a set of basis functions and an applicability with the property that in the case of full certainty, the basis functions coincide with the given correlation kernels. Thus normalized convolution using the computed basis functions and applicability gives the expected results when all data are certain and something that is expected to be useful when uncertain data are present.

By setting $n \times n$ matrices $W_a = diag(a)$, $W_c = diag(c)$, and $W^2 = W_a W_c$ as proposed in [195]. The coefficients u from equation (7.8) will be

$$u = \underbrace{\left[B^T W_a W_c B\right]^{-1}}_{N} \underbrace{B^T W_a W_c t}_{D} \tag{7.11}$$

To solve this problem we first collect the correlation kernels $\{h_i\}$ into the matrix H. With the assumption that the images have full certainty, it means that W_c is an identity-one matrix, this leads to

$$B = W_a^{-1} H (H^T W_a^{-1} H)^{-1} \tag{7.12}$$

An additional requirement is that the correlation kernels are linearly independent, so that $H^T W_a^{-1} H$ is invertible. Now that we have this B, normalized convolution is computed as usual with $W^2 = W_c$ and the expansion coefficients give the filter responses. Obviously we must use the same applicability as in the construction of B.

7.2.2 Fisher Vectors

This section describes the Fisher Vector (FV) of [196]. The FV is an image representation obtained by pooling local image features. It is frequently used as a global image descriptor in visual classification. Given an input image I, the Fisher Vector (FV) formulation of [196] starts by extracting dense-and-multi-scale local descriptors $X = \{x_t, t = 1 \cdots T\}$. It is assumed that X can be modeled by a probability density function u_λ, with λ is the set of parameters of u as well as the estimation of those. Then, X can be described by the gradient vector as

$$G_X^\lambda = \frac{1}{T} \bigtriangledown_\lambda \log u_\lambda(X) \tag{7.13}$$

Since Fisher information matrix (F_λ) of probability density function (u_λ) is symmetric and positive, it has the Cholesky decomposition $F_\lambda = L'_\lambda L_\lambda$. Therefore, the Fisher kernel can be written as a dot-product between normalized gradient vectors Γ_λ, with $\Gamma^X_\lambda = L_\lambda G^X_\lambda$, and Γ^X_λ is referred to as the Fisher vector of X.

The probability density function u_λ is chosen to be a Gaussian Mixture Model (GMM): $u_\lambda(x) = \sum_{i=1}^{K} w_i u_i(x)$, and $\lambda = \{w_i, \mu_i, \Sigma_i, i = 1 \cdots K\}$ where w_i, μ_i, and Σ_i are mixture weight, mean vector and covariance matrix of Gaussian u_i respectively. Covariance matrices are assumed to be diagonal, and σ_i^2 denotes the variance vector. The local descriptors x_t are assumed to be generated independently from u_λ, and therefore:

$$G^X_\lambda = \frac{1}{T} \sum_{t=1}^{T} \bigtriangledown_\lambda \log u_\lambda(x_t). \tag{7.14}$$

Soft assignment (γ_t) of descriptor x_t to Gaussian i is computed as

$$\gamma_t(i) = \frac{w_i u_i(x_t)}{\sum_{j=1}^{K} w_j u_j(x_t)}. \tag{7.15}$$

Let $\Gamma^X_{\mu,i}$ be the gradient vector computed on the mean μ_i, and $\Gamma^X_{\sigma,i}$ be the gradient vector computed on the standard deviation σ_i of Gaussian i, then derivative of those will lead to formulations for computing the first and second order statistics of the local descriptors.

$$\Gamma^X_{\mu,i} = \frac{1}{T\sqrt{w_i}} \sum_{t=1}^{T} \gamma_t(i) \left(\frac{x_t - \mu_i}{\sigma_i} \right). \tag{7.16}$$

$$\Gamma^X_{\sigma,i} = \frac{1}{T\sqrt{2w_i}} \sum_{t=1}^{T} \gamma_t(i) \left[\frac{(x_t - \mu_i)^2}{\sigma_i^2} - 1 \right]. \tag{7.17}$$

The final Fisher vector is the concatenation of the $\Gamma^X_{\mu,i}$ and $\Gamma^X_{\sigma,i}$ vectors for $i = 1...K$, with the dimensionality is $2KD$, D is the dimension of a descriptor x_t.

7.2.3 Normalized-convolution network

This section introduces a handcrafted network which inherits ScatNet [35] for texture classification. In this novel handcrafted network, so called Normalized-Convolution Network (NmzNet), we propose two important changes to the ScatNet (Chapter 4) to enhance its robustness.

1) We modify the wavelet modulus operator, the fundamental operator of ScatNet, by a substitution of the normalized convolution presented in section 7.2.1 for the standard convolution used in ScatNet as follows

The operator of the first network layer:

$$W_1 x = \left(x \circledast \phi_J, \{ |x \circledast \psi_{\theta,j}| \}_{\theta,j} \right) = (S_0 x, U_1 x) \tag{7.18}$$

where $\circledast$ is the normalized convolution operator.
$|.|$ is the modulus operator.
$j = 1...J$ is the scaling parameter.
ϕ_J and $\psi_{\theta,j}$ are respectively the low and high pass filter kernels with J scales and θ angles. These kernels used as applicability functions for the normalized convolution in our proposed method.

The wavelet-modulus operator for layers $m \geq 2$ computed on $U_{m-1}x$ the same way as the one on the first layer. However, there is a normalized convolution computed along the angle parameters taken into account.

2) The concatenation of mean value of ScatNet coefficients is replaced by the Fish Vector feature aggregation in the NmzNet.

Given an image I, local descriptors $\{d_1, ..., d_n\}$ are extracted from three different layers of NmzNet. These features are then soft-quantized using a Gaussian Mixture Model (GMM). Subsequently, the dimensionality is reduced by PCA before concatenating their first and second order statistics to form the final Fisher Vector features.

7.3 Experimental validation

This section evaluates the Normalize-Convolution Network for classifying texture data. First, parameter settings and datasets are presented. Second, we evaluate the results and compare with the state-of-the-art. Finally, we analyze the proposed method and its complexity. In our experiments, we used source code of ScatNet [35], and the implementation of Fisher Vector aggregation in [22] to generate classification results on three challenging texture datasets: UIUC [52], KTH-TIPS-2a, and KTH-TIPS-2b [173].

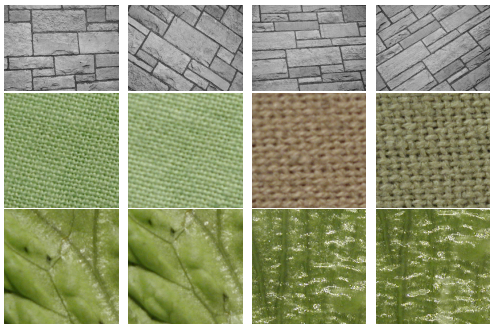


Figure 7.3: Images in the same class from the 3 experimented datasets: Rows 1) UIUC, 2) KTH-TIPS-2a, 3) KTH-TIPS-2b.

7.3.1 Experimental settings

This section discusses how the Normalize-Convolution Network is implemented and enhances texture representation. As illustrated in (Figure 7.4), the first row presents

(from left to right) a Gaussian window used as applicability function, the real and imaginary part of one Gabor filter, windowed filters respectively which can be used for computing normalized convolution. The second row is a lettuce leaf image and its corresponding convolution results. It is obvious that normalized convolution results (right) retain more details from the image than those of standard convolution. This clue inspires us to build a convolution handcrafted network based on ScatNet with normalized convolution used instead of standard convolution.

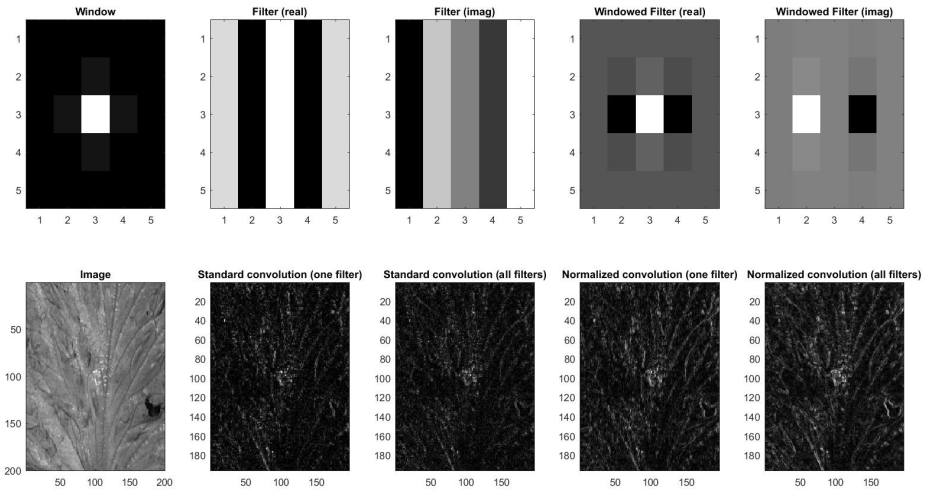


Figure 7.4: Normalized Convolution compared with Standard Convolution, lettuce leaf image from KTH-TIPS-2b dataset

We built Normalized-Convolution Network successively cascading wavelet transform and modulus nonlinear operators, this is inspired by ScatNet model [35] with a major difference is that in our network, normalized convolution is used in stead of standard convolution. Figure 7.5 illustrates images taken from the KTH-TIPS-2b dataset, the two images are in the same class (cork). We extracted feature vectors of these images, using Normalized-Convolution Network. Next we compute χ^2 distance between those, it is 0.45. The same procedures are done with ScatNet, and the χ^2 distance is 14.89. It is obvious that our approach has a better discriminative representation compared with ScatNet.

Finally, we use off-the-shelf method Fisher vector as an aggregation method for our approach. Classification results are presented in the next section with SVM classifier is used.

The parameters of Normalized-Convolution Network are selected such that scaling number is $J = 3$ for datasets with resolution of images at 150×150 or below, $J = 4$ for image size 300×300 or smaller, and $J = 5$ otherwise. Here, the principle is that the smaller image size, the smaller scaling number chosen as recommended for ScatNet in [35]. Orientations of filter bank is set to 8 ($L = 8$), the number of ScatNet layers is set to 3 ($M = 3$). Since if the layer number exceeds this threshold,

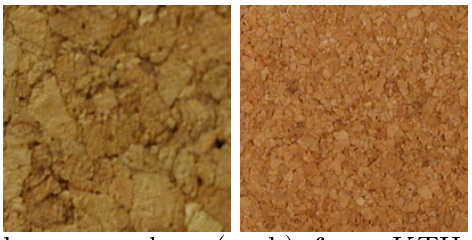


Figure 7.5: Images in the same class (cork) from KTH-TIPS-2b dataset whose NmzNet descriptors have smaller χ^2 distances compared with those of ScatNet.

both feature dimension and feature extraction time increase with minor improvement in classification accuracy. The reason for choosing those parameters is that our model is strongly inspired by ScatNet model with the two important modification as mentioned, we use normalized convolution for the fundamental operator and Fisher vector as aggregation method. Our novel convolution network can be considered as the first variant of ScatNet. It has the full advantages of ScatNet which has been mathematically proved to have the capability to deal with affine transform of texture data. Furthermore, the novel ScatNet variant we proposed has the stronger discriminative power on the very challenging datasets which will be discussed in the next sections.

The following sections will discuss about classification results of our Normalized-Convolution Network and compare them with those of recent state-of-the-arts. Experiments were conducted on three texture datasets, the results are compared with well-known and state-of-the-art of those, and we chose the highest results reported by relevant articles for the comparison. Different form the our two other contribution, the classifier we use here in this chapter is a multi-class SVM classifier rather than a Nearest Neighbor applied Chapter 5 or PCA classifier in Chapter 6 for a fair comparison with state-of-the-arts.

7.3.2 Results on the UIUC database

Classification results on UIUC dataset (Table 7.1) shows that our proposed Normalized-Convolution Network improves the discriminative power slightly when comparing with its predecessor, ScatNet, 96.97 % compared to 96.15 %. Even though, the improvement is significant on the more challenging database such as KTH-TIPS2 presented in the next sections. There is such an enhancement because we propose to use within ScatNet model the normalize-convolution operator [21]. In addition to that, Fisher Vector aggregation method also shows its advantages when aggregating the Normalized-Convolution Network coefficients. They are altogether complementary to form good convolution network for texture representation.

Results in Table 7.1 shows that our accurate rate on UIUC dataset is similar to that of works in [191], roundly 1% lower than the recent frontier FV-SIFT+ FC+FV-VD [179] of this dataset while beating all others. Compared with our previous approaches for texture in this dataset, the proposed Network (NmzNet)

Table 7.1: Classification accuracy comparisions with state-of-the-art on the UIUC database. ($\diamond$), ($\diamond\diamond$), and ($\star$): Our proposed methods in Chapter 5, 6, and this chapter respectively.

Method	Accuracy(%)
NmzNet ($\star$)	96.97
CLEBP ($\diamond$)	95.81
CLBP_S/M+ScatNet ($\diamond\diamond$)	97.20
$MDLBP^{riu2}$ [197]	88.05
LBPHF [144]	89.58
Hayman *et al.*[190]	92.00
CLBPHF [198]	92.55
MFS[193]	92.74
BRINT [33]	93.30
MRELBP (SVM) [34]	96.88
ScatNet(PCA) [35]	96.15
VZ-joint[174]	97.83
OTF[192]	97.40
scLBP [134]	98.45
FV-AlexNet (SVM) [179]	99.1
FV-VGGM (SVM) [179]	99.7
FV-VGGVD (SVM) [179]	99.8

has above 1% more accurate than CLEBP (at 95.81%) while being outweighed by CLBP_S/M+ScatNet (at 97.20%). However, it should be noticed that in this experiment, results presented in Table 7.1, only a single type of features extracted from NmzNet is classified whilst FV-VGGVD (SVM) [179] and CLBP_S/M+ScatNet (our proposed method presented in Chapter 6) are in fact the compound features, i.e. their features are formed by a combination of two or more type of features which are complementary. Recalling that FV-SIFT+ FC+FV-VD [179] combines SIFT features and those extracted from a pre-trained CNN (VGG), and CLBP_S/M+ScatNet is the mixture of LBP and ScatNet features. On contrary, in this experiment, we choose the classification results from only a single type of features, the fisher vector aggregation of Normalized-Convolution Network coefficients, to compare. This is because we want to introduce Normalize-Convolution Network as the original version of our work or the first ScatNet variant.

7.3.3 Results on KTH-TIPS-2a

KTH-TIPS-2a [173] is a very challenging dataset for the texture classification task. State-of-the-art performance on this dataset was reported approximately 78%. KTH-TIPS-2a is challenging because it has very intense intra-class variations, as evident in Figure 7.3. Table 7.2 shows the experimental results obtained on KTH-TIPS-2a. NmzScat achieved a 82.37% classification rate, which was superior to the results obtained by handcrafted conventional methods with a wide margin (being roughly 20% different from VZ-joint[174], and 4% from the scLBP [134], the best

LBP variant performance reported on this dataset). The classification accurate rate of NmzScat is slightly better than those of our framework proposed in Chapter 6 (BF+CLBP+ScatNet, 82.05%) while it is much better than our LBP variant presented in Chapter 5 (CLEBP, 74.50%). NmzScat has a significant classification result on this dataset, far outweighs CLEBP which has very high performance on Outex due to the fact that Normalized-Convolution [21] is a decent operator to be used in a three-layer convolution-modulus network. In addition, Fisher Vector aggregation also contributes to the success of our Normalized-Convolution Network, the first ScatNet variant has ever reported.

Table 7.2: Classification accuracy comparisons with state-of-the-art on the KTH-TIPS-2a database. ($\diamond$), ($\diamond\diamond$), and ($\star$): Our proposed methods in Chapter 5, 6, and this chapter respectively.

Method	Accuracy(%)
NmzNet ($\star$)	82.37
CLEBP ($\diamond$)	74.50
BF+CLBP_S/M+ScatNet ($\diamond\diamond$)	82.05
LBP [69]	58.10
CLBP [133]	66.58
VZ-MR8 [176]	62.35
VZ-joint[174]	61.93
Multi-Scale-BIF [189]	71.56
ScatNet [35]	72.57
COV-LBPD [185]	74.86
MLEP [139]	75.57
scLBP [134]	78.39

7.3.4 Results on KTH-TIPS-2b

For the classification setting on KTHTIPS2b texture dataset, as a protocol we follow the scheme used in [173]. That is experiment conducted by training on one, two, or three samples; testing is always conducted only on un-seen ones. Thus, there are six experiments in which two samples are used as training and the two others are used as testing, and the classification rate reported is the average of those results. In this challenging database, our the proposed convolution network demonstrates its superior strength to other handcrafted features including our proposed methods, they are the CLEBP presented in Chapter 5 and BF+CLBP+ScatNet framework introduced in Chapter 6. Table 7.3 details the results obtained on this challenging dataset and compares accurate gain of the proposed network with other recent state-of-the-art methods. As can be seen in the table, our approach (at the classification accuracy 78.10 %) considerably outperforms all competing handcrafted algorithms. It is only inferior to the 19-layer-CNN and SIFT hybrid method with Fisher Vector aggregation, the FV-SIFT+ FC+FV-VD[179] in term of classification accuracy (81.50 %). It is worth noticing that our handcrafted network has only three convolution-modulus

Table 7.3: Classification accuracy comparisons with state-of-the-art on the KTH-TIPS-2b database. ($\diamond$), ($\diamond\diamond$), and ($\star$): Our proposed methods in Chapter 5, 6, and this chapter respectively.

Method	Accuracy(%)
NmzNet ($\star$)	78.10
CLEBP ($\diamond$)	70.50
BF+CLBP_S/M+ScatNet ($\diamond\diamond$)	78.09
VZ-MR8[176]	55.70
NRLBP[186]	57.00
NTLBP[131]	58.78
MBP[187]	60.29
LBP [69]	60.35
VZ-Path [174]	60.70
LTP[70]	62.12
PRICoLBP[135]	61.17
COV-LBPD[185]	63.47
MWLD[151]	64.70
ELBP[74]	64.84
MSJLBP[136]	65.51
$MDLBP^{riu2}$[197]	66.52
BRINT[33]	66.67
LBPHF[144]	67.51
CLBPHF[198]	68.10
ScatNet(PCA)[35]	68.92
MRELBP[34]	68.98
MRELBP (SVM)[34]	77.91
FV-AlexNet (SVM) [179]	77.9
FV-SIFT+ FC+FV-VD (SVM)[179]	81.50

layers. The best results of the highest performance (our proposed compound framework presented in Chapter 6) among handcrafted descriptors is 78.09% while the result of our novel convolution network is 78.10 %. This performance is much better than ScatNet (68.92 %), the network from which our proposed network is derived.

As can be seen from Table 7.3, our proposed network gets a significant improvement over the original features it inherits from, the enhancement in classification accuracy is approximately 9%. Also, our classification result on this database is ranked closely to FV-SIFT+ FC+FV-VD [179]. As discussed earlier, the method of our third contribution presented in this chapter can demonstrate its superior strength to other handcrafted features because it can leverage strength of Normalized-Convolution operator [21], ScatNet architecture, and Fisher Vector aggregation method. Normalized-Convolution operator and Normalized-Convolution with the theory of signal/certainty was proved mathematically to be sufficient operator for dealing with degraded images, and Fisher Vector shows its power when applying as an aggregation method of FV-SIFT+ FC+FV-VD (SVM)[179].

Although our classification result on this database is inferior to FV-SIFT+ FC+FV-VD [179], it is on the high ranking.

7.4 Conclusions

In this chapter, we have proposed the Normalized-Convolution Network (NmzNet), a handcrafted convolution network which leverages off-the-shelf methods such as Normalized Convolution, ScatNet, Fisher Vector aggregation for efficient texture classification. Our approach extracts features which are robust to scale, rotation, and illumination changes.

Future works can be drawn on other categories of texture data such as noise and occlusion data with other types of feature aggregation. Next chapter of this thesis will draw a perspective of this network in a more detail.

Chapter 8

Conclusions

This thesis started by providing definition of texture, texture analysis and its applications followed by a literature review. Perhaps, it is therefore appropriate to end by looking at some of the applications that our work has been put to and reviewing extensions of it in the literature. Finally, possible directions of future research will be also discussed.

8.1 Applications

It is obvious that texture analysis in general and texture classification in particular has a great variety of applications. As such, the most mentioned domains include textile industry, image retrieval, medical imaging, remote sensing, and so on. However, we want to figure out in this section some very practical applications in which our works can be applied on such as industrial inspection, and detecting face spoofing attacks in a authentication security system based on biometric measures.

A good example of the inspection system is introduced in [199]. In this machining process, a computer system controls the movement and operation of the mills, lathes, and other cutting machines, and images of the workpiece surfaces are analyzed to distinguish sharp tools from the blunt ones or to recognize the output products as good ones or faults using machine vision techniques. Figure 8.1 illustrates a machining system whose operations are monitored based on surface texture and vibration signal.

In the monitoring platform illustrated in Figure 8.1, along every cutter path, 10 cutting positions are set with a interval of $10mm$ for every two of them. During the milling process, a piece of 10-second vibration signal is acquired for every cutting position which is machined. The surface texture is inspected by using the microscope under a fixed distance and illumination condition. A surface texture image is

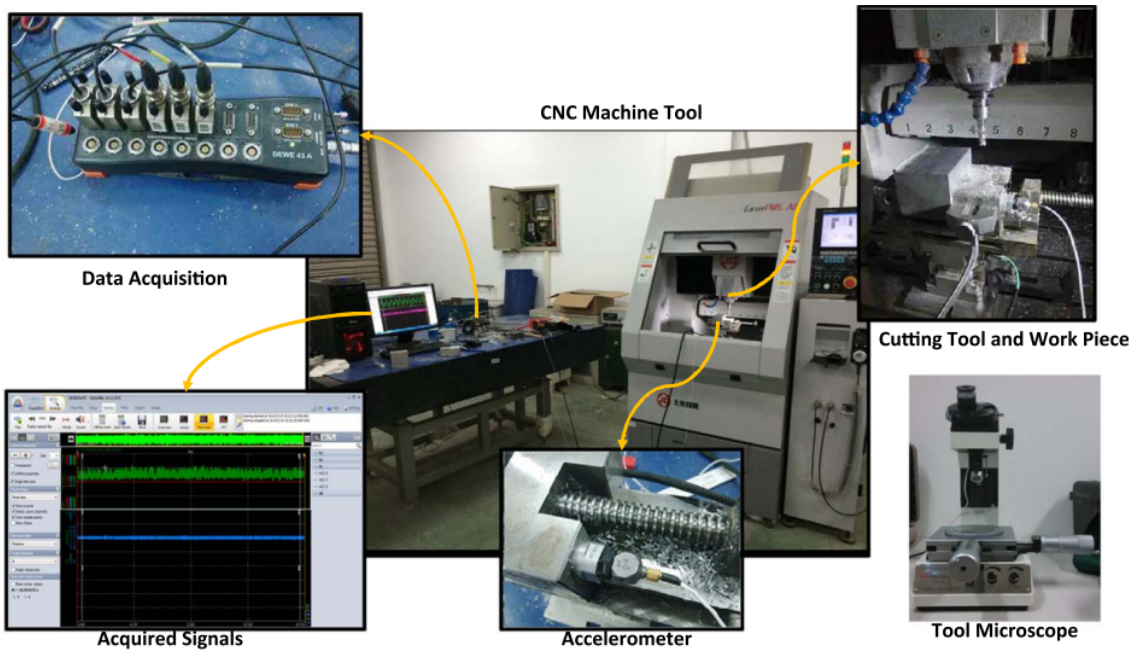


Figure 8.1: A monitoring platform used in [199].

captured from a $2mm \times 2mm$ area at every cutting position. Both surface texture image and vibration signal are collected for every cutting position. Finally, 180 data group are obtained, and both up-milling and down-milling are considered.

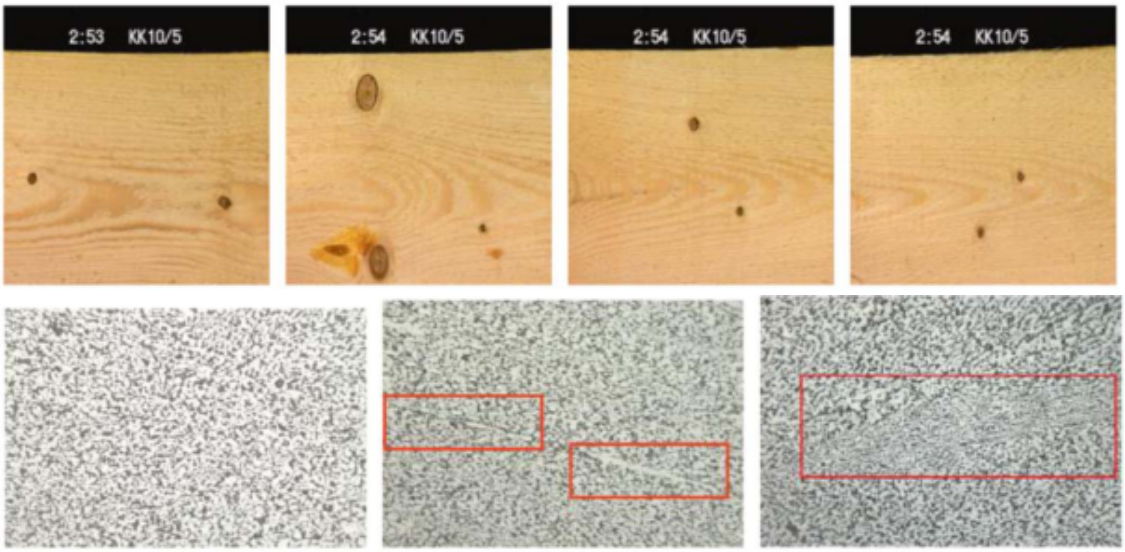


Figure 8.2: Wood and micro-structure defect samples used in [200].

Surface texture inspection using texture features is also type of automated surface inspection. It is an important but challenging domain in the industry. Figure 8.2 presents data samples used in [200] where texture features are applied to detect fault materials. Since collecting training data is usually costly, for this reason training data can be limited. Therefore, surface inspection should deal well with the lack of training data cases. Another factor should be aware of is that related learning

methods are highly data-dependent when amount of data for training is limited. In this certain situation, CNN-based methods in general may face challenges while our handcrafted feature methods developed in Chapter 5, 6, and 7 can be useful.

In addition, CNN-based method such as framework proposed in [200] has high inspection performance in comparison with other handcrafted methods. However, it is claimed in the literature that it needs further investigation for the real-time inspection applications. On the contrary, our novel LBP variant proposed in Chapter 5 can be a potential measure for this situation.

Another striking example of applying texture classification is detecting face spoofing attacks in a authentication system based on biometric security measures. When being spoofed, a biometric recognition system is bypassed by presenting a copy of the biometric evidence of a valid user. Among several type of biometric modalities, spoofing a face recognition system is particularly easy to perform: all its need is simply a photograph of a valid user.

It is not easy for human eye to distinguish an image of a real user and a recaptured image of the same user at a very first glance (Figure 8.3). However, there is a certain difference between the real face and spoof-attack images when they are translated into a proper feature space. The disparities come from the fact that the human face as a 3D object, as well as the human skin, have their own optical qualities (absorption, reflection, scattering, refraction), which other materials (paper, photographic paper or electronic display) do not possess. Therefore, the texture properties of real accesses and spoof-attacks will be different. Chingovska *et al.*[201] address the problem of detecting face spoofing attacks by inspecting the potential of texture features and their variations on types of attacks: printed photographs, and photos and videos displayed on electronic screens of different sizes. They exploit the capabilities of LBP texture features to reveal the difference between real accesses and attacks with photographs and videos. It is obvious that our proposed methods presented in Chapter 5, 6, 7 are superior to the original LBP texture features used in [201] can be applied in the same application domain with potentially competitive performances.

8.2 Conclusions

This doctoral dissertation proposes many complementary approaches for efficient texture classification.

Here are some of the key contributions made in the thesis. The first contribution of this work is the development of a novel LBP variant in Chapter 5, called the completed local entropy binary patterns (CLEBP), for texture presentation. The approach is developed by applying the LBP-based self-similarity operators on the image entropy space. This helps to incorporate the complementary information, entropy, into LBP-based features to extract both micro and macro structure information of images. Consequently, CLEBP and the conventional LBP have complementary strength and that by combining these algorithms, one obtains better results than either of them considered separately. Experimental results on four large

Figure 8.3: The order from left to right and top to bottom, images show examples of real access, printed photograph, mobile phone and tablet attacks respectively.

texture databases, including Outex, KTHTIPS2b, CuRet and UIUC show that our approach is more efficient than many other LBP variants. Although we have incorporated the information quantity (a type of macro structure information) into LBP features to compensate for macro structure information loss of micro LBP features, the supplementary components are formed by the same type of micro-feature LBP encoding method. This results in high performance on texture data which is less or not variant on scales such as Outex while having poor performance on datasets like KTH-TIPS2. The next contribution introduces an alternative approach texture representation, a compound framework formed by combining two complementary types of features results in high discriminative texture method.

The second contribution is the proposed framework in which BF+CLBP, and its supplementary features (ScatNet) are combined, and PCA classifier used. This is detailed in Chapter 6. The LBP operator [29] is a theoretically simple yet a ef-

ficient method of analyzing textures. It unifies the traditional divergent statistical and structural models of texture analysis. LBP method describes texture in terms of micro-primitives (textons), and their statistical placement rules. It also has measures to deal with the rotation and multi-scale problems of texture. In addition, CLBP[24] is a variant which is close related but more robust than the original version. The power of this can be enhanced when using the complementary BF preprocessing technique [102]. Even though, this suite of measures only captures the micro-primitives while losing the macro-primitives because this traditional LBP method basically thresholds the neighborhood against its center in a certain radius. Another shortcoming of LBP method is the way it copes with the multi-scale issue in such a way that will lead to a high dimension of feature vector when using up to nine scales. Those drawbacks can be removed when LBP is combined with ScatNet [35]. The empirical study in this thesis and related works show that CLBP and ScatNet are not concurrent yet complementary due to their several properties. First, ScatNet catches a wider range primitives in comparison with LBP because ScatNet cascades multi-scale wavelet transform operators for its implementation. The macro-primitives captured by ScatNet can compensate for the lost of LBP method. Second, ScatNet extracts discriminative multi-scale features while keeping the feature vector very compact, dimension is 536 (this number is approximately 1800 for nine-scale LBP) if ScatNet features are computed over image patches of size $2^J = 2^5 = 32$ with $K = 8$ wavelet orientations. Because ScatNet is proved mathematically in [160] that it is invariant to rotation, translation and small deformation, and we experimentally show in the thesis that these properties supplement to LBP when they are combined. Finally, PCA classifier is integrated to form a robust and efficient framework for texture classification. It should be noticed that CLBP (a basic LBP method) rather than CLEBP (a novel advance method) and ScatNet have been combined because we want to prove in this contribution that a basic micro LBP and macro structure feature ScatNet can be complementary. Advanced LBPs combined with ScatNet is intended for future works. With an alternative approach, next contribution introduces a novel ScatNet variant which enhances the strength of ScatNet for texture representation.

The third contribution is presented in Chapter 7. In the chapter we propose a novel handcrafted convolution network, so-called Normalized-Convolution Network. Our model in this work is inspired by the ScatNet model [20] with two important modifications. 1) Normalized Convolution operator [21] substitute for the standard convolution. Normalized Convolution is based on the the signal/certainty philosophy, i.e. separating the values of a signal from the certainty of the measurements. 2) The implementation [22] of Fisher Vector aggregation of the network coefficients is applied instead of the mean values of those. The proposed method not only gains high competitive results on the challenging texture databases: UIUC, KTH-TIPS2 but also poses the potentiality to high performance on other challenging texture data such as occlusion or noised data.

8.3 Perspective

The possibilities to extend our complementary approaches and improvements of the algorithms have been discussed throughout the thesis. In addition to those there are some more ideas we want to mention.

The LBP-ScatNet combination framework has the potentiality of noise tolerant since it is constituted by BF+CLBP and ScatNet which are proved to be robust to noise. However, this needs to have further experiment and test on noise data.

Normalized Convolution Network uses the normalized convolution as basic operator. This operator has the signal/certainty principle and the weighted least square theory behind, and is proved as a powerful measure to deal with degraded data. Therefore, there is a potential that our Normalized Convolution Network can have good representation of degraded and occlusive texture images. This is worth to be explored in the future works. Formulating a model for extracting features from the degraded, and occlusive texture using Normalized Convolution Network can be a interesting work.

More recent, Oyallon *et al.* [68] proposed to use Scattering Network, a handcrafted method, as a hybrid manner with deep CNNs. Additionally, Oyallon *et al.* [68] also show that early layers of deep CNNs do not necessarily need to be learned. Instead, they used pre-defined parameters from a Garbor filter bank. Their work has the best results to-date in comparison with pre-defined representations, and also gains competitive performance compared to Deep CNNs while having lower number of network layers.

Our Normalized Convolution Network [194] shares several common characteristics with Scattering Network. Thus, the pre-defined features of this handcrafted network can provide a parameter saving measure on deep learning techniques and to allow them to require lower resource in both computation and training data.

To conclude, in the CNN and deep learning era, handcrafted features still have their roles to play due to several reasons. First of all, handcrafted descriptors are inspired by the learning features for computational savings. For example, Local Binary Pattern [29] is inspired in a work related to CNN [202], called Local Binary Convolutional Neural Networks (LBCNN) which aims at statistical and computation efficiency while having CNN-comparable performances. Or in another case, a handcrafted feature method, SIFT [11] is combined with filter banks learned from VGG CNN in [8] to enhance classification performance, and this work has state-of-the-art results on many texture datasets. Being different from the other two works mentioned, Scattering Network hybridizes with CNNs in [68] for network-layer reduction while the hybrid method still has decent approximation results of a full layer CNN. Last but not least, in some domains, collecting training dataset is usually costly and related methods are highly dataset-dependent. Our works on handcrafted texture features therefore to pursuit similar targets of those.

Bibliography

[1] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi, "Describing textures in the wild," in *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[2] R. M. Haralick, "Statistical and structural approaches to texture," *Proceedings of the IEEE*, vol. 67, no. 5, pp. 786–804, 1979.

[3] G. R. Cross and A. K. Jain, "Markov random field texture models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, no. 1, pp. 25–39, 1983.

[4] J. R. Bergen and M. S. Landy, "Computational modeling of visual texture segregation," *Computational models of visual processing*, vol. 1, pp. 253–271, 1991.

[5] B. Julesz, "Textons, the elements of texture perception, and their interactions," *Nature*, vol. 290, no. 5802, pp. 91–97, 1981.

[6] B. Julesz, E. Gilbert, L. Shepp, and H. Frisch, "Inability of humans to discriminate between visual textures that agree in second-order statistics?revisited," *Perception*, vol. 2, no. 4, pp. 391–405, 1973.

[7] J. G. Daugman, "Entropy reduction and decorrelation in visual coding by oriented neural receptive fields," *IEEE Transactions on Biomedical Engineering*, vol. 36, no. 1, pp. 107–114, 1989.

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

[9] T. Ojala, T. Maenpaa, M. Pietikainen, J. Viertola, J. Kyllonen, and S. Huovinen, "Outex-new framework for empirical evaluation of texture analysis algorithms," in *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, vol. 1. IEEE, 2002, pp. 701–706.

[10] L. Liu, P. Fieguth, Y. Guo, X. Wang, and M. Pietikäinen, "Local binary features for texture classification: Taxonomy and experimental study," *Pattern Recognition*, vol. 62, pp. 135–160, 2017.

[11] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[12] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[13] S. Rao, H. Mobahi, A. Yang, S. Sastry, and Y. Ma, "Natural image segmentation with adaptive texture and boundary encoding," *Computer Vision–ACCV 2009*, pp. 135–146, 2010.

[14] A. A. Efros and W. T. Freeman, "Image quilting for texture synthesis and transfer," in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. ACM, 2001, pp. 341–346.

[15] A. Criminisi, P. Pérez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," *IEEE Transactions on image processing*, vol. 13, no. 9, pp. 1200–1212, 2004.

[16] M. Clerc, "Weak homogeneity for shape from texture," 2002.

[17] D. A. Forsyth, "Shape from texture without boundaries," in *European Conference on Computer Vision*. Springer, 2002, pp. 225–239.

[18] A. Criminisi and A. Zisserman, "Shape from texture: Homogeneity revisited." in *BMVC*, 2000, pp. 1–10.

[19] L. Nanni, S. Ghidoni, and S. Brahnam, "Handcrafted vs non-handcrafted features for computer vision classification," *Pattern Recognition*, 2017.

[20] J. Bruna and S. Mallat, "Invariant scattering convolution networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1872–1886, 2013.

[21] H. Knutsson and C.-F. Westin, "Normalized and differential convolution," in *Computer Vision and Pattern Recognition, 1993. Proceedings CVPR'93., 1993 IEEE Computer Society Conference on*. IEEE, 1993, pp. 515–523.

[22] P.-H. Gosselin, N. Murray, H. Jégou, and F. Perronnin, "Revisiting the fisher vector for fine-grained classification," *Pattern recognition letters*, vol. 49, pp. 92–98, 2014.

[23] T. Ojala, M. Pietikäinen, and D. Harwood, "A comparative study of texture measures with classification based on featured distributions," *Pattern Recognition*, vol. 29, no. 1, pp. 51–59, 1996.

[24] Z. Guo, L. Zhang, and D. Zhang, "A completed modeling of local binary pattern operator for texture classification," *IEEE Transactions on Image Processing*, vol. 19, no. 6, pp. 1657–1663, Jun. 2010.

[25] B. Julesz, "Visual pattern discrimination," *IRE transactions on Information Theory*, vol. 8, no. 2, pp. 84–92, 1962.

[26] R. M. Haralick, K. Shanmugam *et al.*, "Textural features for image classification," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 3, no. 6, pp. 610–621, 1973.

[27] J. S. Weszka, C. R. Dyer, and A. Rosenfeld, "A comparative study of texture measures for terrain classification," *IEEE Transactions on Systems, Man, and Cybernetics*, no. 4, pp. 269–285, 1976.

[28] T. Randen and J. H. Husøy, "Filtering for texture classification: A comparative study," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 4, pp. 291–310, 1999.

[29] T. Ojala, M. Pietikäinen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 7, pp. 971–987, 2002.

[30] T. Leung and J. Malik, "Representing and recognizing the visual appearance of materials using three-dimensional textons," *International journal of computer vision*, vol. 43, no. 1, pp. 29–44, 2001.

[31] M. Varma and A. Zisserman, "Classifying images of materials: Achieving viewpoint and illumination independence," in *European Conference on Computer Vision*. Springer-Verlag, 2002, pp. 255–271.

[32] P. Saisan, G. Doretto, Y. N. Wu, and S. Soatto, "Dynamic texture recognition," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 2. IEEE, 2001, pp. II–II.

[33] L. Liu, Y. Long, P. W. Fieguth, S. Lao, and G. Zhao, "Brint: binary rotation invariant and noise tolerant texture classification," *IEEE Transactions on Image Processing*, vol. 23, no. 7, pp. 3071–3084, 2014.

[34] L. Liu, S. Lao, P. W. Fieguth, Y. Guo, X. Wang, and M. Pietikäinen, "Median robust extended local binary pattern for texture classification," *IEEE Transactions on Image Processing*, vol. 25, no. 3, pp. 1368–1381, 2016.

[35] L. Sifre and S. Mallat, "Rotation, scaling and deformation invariant scattering for texture discrimination," in *2013 IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1233–1240.

[36] M. Tuceryan, A. K. Jain *et al.*, "Texture analysis," *Handbook of pattern recognition and computer vision*, vol. 2, pp. 235–276, 1993.

[37] J. M. Coggins and A. K. Jain, "A spatial filtering approach to texture analysis," *Pattern recognition letters*, vol. 3, no. 3, pp. 195–203, 1985.

[38] T. Caelli and G. Moraglia, "On the detection of gabor signals and discrimination of gabor textures," *Vision research*, vol. 25, no. 5, pp. 671–684, 1985.

[39] B. Julesz and J. R. Bergen, "Human factors and behavioral science: Textons, the fundamental elements in preattentive vision and perception of textures," *The Bell system technical journal*, vol. 62, no. 6, pp. 1619–1645, 1983.

[40] J. R. Bergen and E. H. Adelson, "Early vision and texture perception," *Nature*, vol. 333, no. 6171, pp. 363–364, 1988.

[41] J. Malik and P. Perona, "Preattentive texture discrimination with early vision mechanisms," *JOSA A*, vol. 7, no. 5, pp. 923–932, 1990.

[42] H. Voorhees and T. Poggio, "Computing texture boundaries from images," *Nature*, vol. 333, no. 6171, pp. 364–367, 1988.

[43] W. T. Freeman, "Steerable filters and local analysis of image structure," MASSACHUSETTS INST OF TECH CAMBRIDGE, Tech. Rep., 1992.

[44] H. Greenspan, S. Belongie, R. Goodman, and P. Perona, "Rotation invariant texture recognition using a steerable pyramid," in *Pattern Recognition, 1994. Vol. 2-Conference B: Computer Vision & Image Processing., Proceedings of the 12th IAPR International. Conference on*, vol. 2. IEEE, 1994, pp. 162–167.

[45] G. M. Haley and B. Manjunath, "Rotation-invariant texture classification using a complete space-frequency model," *IEEE transactions on Image Processing*, vol. 8, no. 2, pp. 255–269, 1999.

[46] P. Brodatz, *Textures: a photographic album for artists and designers*. Dover Pubns, 1966.

[47] K. J. Dana, B. van Ginneken, S. K. Nayar, and J. J. Koenderink, "Reflectance and texture of real-world surfaces," *ACM Trans. Graph.*, vol. 18, no. 1, pp. 1–34, 1999.

[48] S. Konishi and A. L. Yuille, "Statistical cues for domain specific image segmentation with performance analysis," in *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, vol. 1. IEEE, 2000, pp. 125–132.

[49] C. Schmid, "Constructing models for content-based image retrieval," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 2. IEEE, 2001, pp. II–II.

[50] O. G. Cula and K. J. Dana, "3d texture recognition using bidirectional feature histograms," *International Journal of Computer Vision*, vol. 59, no. 1, pp. 33–60, 2004.

[51] F. Schaffalitzky and A. Zisserman, "Viewpoint invariant texture matching and wide baseline stereo," in *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, vol. 2. IEEE, 2001, pp. 636–643.

[52] S. Lazebnik, C. Schmid, and J. Ponce, "A sparse texture representation using local affine regions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, pp. 1265–1278, 2005.

[53] E. Hayman, B. Caputo, M. Fritz, and J.-O. Eklundh, "On the significance of real-world conditions for material classification," *Computer Vision-ECCV 2004*, pp. 253–266, 2004.

[54] M. Varma, "Statistical approaches to texture classification," Ph.D. dissertation, University of Oxford, 2004.

[55] C.-M. Pun and M.-C. Lee, "Log-polar wavelet energy signatures for rotation and scale invariant texture classification," *IEEE transactions on pattern analysis and machine intelligence*, vol. 25, no. 5, pp. 590–603, 2003.

[56] J. G. Daugman, "Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters," *JOSA A*, vol. 2, no. 7, pp. 1160–1169, 1985.

[57] A. Mojsilovic, M. V. Popovic, and D. M. Rackov, "On the selection of an optimal wavelet basis for texture characterization," *IEEE Transactions on Image Processing*, vol. 9, no. 12, pp. 2043–2050, 2000.

[58] S. G. Mallat, "A theory for multiresolution signal decomposition: the wavelet representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 7, pp. 674–693, 1989.

[59] F. Ade, "Characterization of textures by âĂŸeigenfiltersâĂŹ," *Signal Processing*, vol. 5, no. 5, pp. 451–457, 1983.

[60] T. Randen, "Filter and filter bank design for image texture recognition," *Doctoral dissert., Norwegian Univ. of Science and Techn., Stravanger College*, 1997.

[61] S. Z. Li, *Markov random field modeling in image analysis*. Springer Science & Business Media, 2009.

[62] S. C. Zhu, Y. Wu, and D. Mumford, "Filters, random fields and maximum entropy (frame): Towards a unified theory for texture modeling," *International Journal of Computer Vision*, vol. 27, no. 2, pp. 107–126, 1998.

[63] D. Dunn and W. E. Higgins, "Optimal gabor filters for texture segmentation," *IEEE Transactions on image processing*, vol. 4, no. 7, pp. 947–964, 1995.

[64] M. Unser, "Local linear transforms for texture measurements," *Signal processing*, vol. 11, no. 1, pp. 61–79, 1986.

[65] A. Mahalanobis and H. Singh, "Application of correlation filters for texture recognition," *Applied Optics*, vol. 33, no. 11, pp. 2173–2179, 1994.

[66] A. K. Jain and K. Karu, "Learning texture discrimination masks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 2, pp. 195–205, 1996.

[67] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[68] E. Oyallon, E. Belilovsky, and S. Zagoruyko, "Scaling the scattering transform: Deep hybrid networks," in *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[69] T. Ojala, M. Pietikäinen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2002.

[70] X. Tan and B. Triggs, "Enhanced local texture feature sets for face recognition under difficult lighting conditions," in *Analysis and Modeling of Faces and Gestures*. Springer, 2007, pp. 168–182.

[71] S. Liao, M. W. Law, and A. C. Chung, "Dominant local binary patterns for texture classification," *IEEE Transactions on Image Processing*, vol. 18, no. 5, pp. 1107–1118, 2009.

[72] Z. Guo, L. Zhang, and D. Zhang, "Rotation invariant texture classification using lbp variance (lbpv) with global matching," *Pattern recognition*, vol. 43, no. 3, pp. 706–719, 2010.

[73] X. Qian, X.-S. Hua, P. Chen, and L. Ke, "Plbp: An effective local binary patterns texture descriptor with pyramid representation," *Pattern Recognition*, vol. 44, no. 10, pp. 2502–2515, 2011.

[74] L. Liu, L. Zhao, Y. Long, G. Kuang, and P. Fieguth, "Extended local binary patterns for texture classification," *Image and Vision Computing*, vol. 30, no. 2, pp. 86–99, 2012.

[75] G. Zhao, T. Ahonen, J. Matas, and M. Pietikäinen, "Rotation-invariant image and video description with local binary pattern features," *IEEE Transactions on Image Processing*, vol. 21, no. 4, pp. 1465–1477, 2012.

[76] Y. Guo, G. Zhao, and M. PietikäInen, "Discriminative features for texture description," *Pattern Recognition*, vol. 45, no. 10, pp. 3834–3843, 2012.

[77] A. C. Bovik, M. Clark, and W. S. Geisler, "Multichannel texture analysis using localized spatial filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 1, pp. 55–73, 1990.

[78] R. Porter and N. Canagarajah, "Robust rotation-invariant texture classification: wavelet, gabor filter and gmrf based schemes," *IEE Proceedings-Vision, Image and Signal Processing*, vol. 144, no. 3, pp. 180–188, 1997.

[79] G. M. Haley and B. Manjunath, "Rotation-invariant texture classification using a complete space-frequency model," *IEEE Transactions on Image Processing*, vol. 8, no. 2, pp. 255–269, 1999.

[80] M. Fritz, E. Hayman, B. Caputo, and J.-O. Eklundh, "The kth-tips database," 2004.

[81] J. J. Koenderink, A. J. Van Doorn, K. J. Dana, and S. Nayar, "Bidirectional reflection distribution function of thoroughly pitted surfaces," *International Journal of Computer Vision*, vol. 31, no. 2, pp. 129–144, 1999.

[82] J. J. Koenderink, A. J. Van Doorn, and M. Stavridi, "Bidirectional reflection distribution function expressed in terms of surface scattering modes," in *European Conference on Computer Vision*. Springer, 1996, pp. 28–39.

[83] P. Mallikarjuna, A. T. Targhi, M. Fritz, E. Hayman, B. Caputo, and J.-O. Eklundh, "The kth-tips2 database," *Computational Vision and Active Perception Laboratory (CVAP), Stockholm, Sweden*, 2006.

[84] M. Pietikäinen, T. Ojala, and Z. Xu, "Rotation-invariant texture classification using feature distributions," *Pattern Recognition*, vol. 33, no. 1, pp. 43–52, 2000.

[85] H. Zhou, R. Wang, and C. Wang, "A novel extended local-binary-pattern operator for texture analysis," *Information Sciences*, vol. 178, no. 22, pp. 4314–4325, 2008.

[86] T. Mäenpää, *The local binary pattern approach to texture analysis: extensions and applications*. Oulun yliopisto, 2003.

[87] M. Pietikäinen, A. Hadid, G. Zhao, and T. Ahonen, "Local binary patterns for still images," in *Computer vision using local binary patterns*. Springer, 2011, pp. 13–47.

[88] F. Tomita, Y. Shirai, and S. Tsuji, "Description of textures by a structural analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, no. 2, pp. 183–191, 1982.

[89] R. W. Conners and C. A. Harlow, "A theoretical comparison of texture algorithms," *IEEE transactions on pattern analysis and machine intelligence*, no. 3, pp. 204–222, 1980.

[90] K. Valkealahti and E. Oja, "Reduced multidimensional co-occurrence histograms in texture classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 1, pp. 90–94, 1998.

[91] T. Ojala, K. Valkealahti, E. Oja, and M. Pietikäinen, "Texture discrimination with multidimensional distributions of signed gray-level differences," *Pattern Recognition*, vol. 34, no. 3, pp. 727–739, 2001.

[92] D.-C. He and L. Wang, "Texture unit, texture spectrum, and texture analysis," *IEEE transactions on Geoscience and Remote Sensing*, vol. 28, no. 4, pp. 509–512, 1990.

[93] L. Wang and D.-C. He, "Texture classification using texture spectrum," *Pattern Recognition*, vol. 23, no. 8, pp. 905–910, 1990.

[94] I. Aleksander and T. J. Stonham, "Guide to pattern recognition using random-access memories," *IEE Journal on Computers and Digital Techniques*, vol. 2, no. 1, pp. 29–40, 1979.

[95] D. Patel and T. Stonham, "A single layer neural network for texture discrimination," in *Circuits and Systems, 1991., IEEE International Sympoisum on.* IEEE, 1991, pp. 2656–2660.

[96] ——, "Texture image classification and segmentation using rank-order clustering," in *Pattern Recognition, 1992. Vol. III. Conference C: Image, Speech and Signal Analysis, Proceedings., 11th IAPR International Conference on.* IEEE, 1992, pp. 92–95.

[97] L. Hepplewhite and T. Stonhamm, "Texture classification using n-tuple pattern recognition," in *Pattern Recognition, 1996., Proceedings of the 13th International Conference on*, vol. 4. IEEE, 1996, pp. 159–163.

[98] M. Varma and A. Zisserman, "Classifying materials from images: to cluster or not to cluster." ECCV, 2002.

[99] M. Pietikäinen, T. Nurmela, T. Mäenpää, and M. Turtinen, "View-based recognition of real-world textures," *Pattern Recognition*, vol. 37, no. 2, pp. 313–323, 2004.

[100] R. E. Sánchez-Yáñez, E. V. Kurmyshev, and F. J. Cuevas, "A framework for texture classification using the coordinated clusters representation," *Pattern Recognition Letters*, vol. 24, no. 1, pp. 21–31, 2003.

[101] W. Zhang, S. Shan, W. Gao, X. Chen, and H. Zhang, "Local gabor binary pattern histogram sequence (lgbphs): A novel non-statistical model for face representation and recognition," in *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, vol. 1. IEEE, 2005, pp. 786–791.

[102] N.-S. Vu, T. P. Nguyen, and C. Garcia, "Improving texture categorization with biologically-inspired filtering," *Image Vision Comput.*, vol. 32, no. 6-7, pp. 424–436, 2014.

[103] C.-H. Yao and S.-Y. Chen, "Retrieval of translated, rotated and scaled color textures," *Pattern Recognition*, vol. 36, no. 4, pp. 913–929, 2003.

[104] N.-S. Vu and A. Caplier, "Face recognition with patterns of oriented edge magnitudes," in *ECCV*, 2010, pp. 313–326. [Online]. Available: `http://www.springerlink.com/content/k510660437327600/`

[105] S. Liao and A. C. Chung, "Face recognition by using elongated local binary patterns with average maximum distance gradient magnitude," in *Asian conference on computer vision*. Springer, 2007, pp. 672–679.

[106] L. Nanni, A. Lumini, and S. Brahnam, "Local binary patterns variants as texture descriptors for medical image analysis," *Artificial intelligence in medicine*, vol. 49, no. 2, pp. 117–125, 2010.

[107] A. Petpon and S. Srisuk, "Face recognition with local line binary pattern," in *Image and Graphics, 2009. ICIG'09. Fifth International Conference on*. IEEE, 2009, pp. 533–539.

[108] S. ul Hussain and B. Triggs, "Visual recognition using local quantized patterns," in *Computer Vision–ECCV 2012*. Springer, 2012, pp. 716–729.

[109] M. Heikkilä, M. Pietikäinen, and C. Schmid, "Description of interest regions with local binary patterns," *Pattern recognition*, vol. 42, no. 3, pp. 425–436, 2009.

[110] A. Fernández, M. X. Álvarez, and F. Bianconi, "Image classification with binary gradient contours," *Optics and Lasers in Engineering*, vol. 49, no. 9, pp. 1177–1184, 2011.

[111] J. Trefnỳ and J. Matas, "Extended set of local binary patterns for rapid object detection," in *Computer Vision Winter Workshop*, 2010, pp. 1–7.

[112] K. Wang, C.-E. Bichot, C. Zhu, and B. Li, "Pixel to patch sampling structure and local neighboring intensity relationship patterns for texture classification," *IEEE Signal Processing Letters*, vol. 20, no. 9, pp. 853–856, 2013.

[113] L. Wolf, T. Hassner, and Y. Taigman, "Effective unconstrained face recognition by combining multiple descriptors and learned background statistics," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 10, pp. 1978–1990, 2011.

[114] N.-S. Vu and A. Caplier, "Enhanced patterns of oriented edge magnitudes for face recognition and image matching," *IEEE Transactions on Image Processing*, vol. 21, no. 3, pp. 1352–1365, 2012.

[115] M. Calonder, V. Lepetit, M. Ozuysal, T. Trzcinski, C. Strecha, and P. Fua, "Brief: Computing a local binary descriptor very fast," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 7, pp. 1281–1298, 2012.

[116] S. Leutenegger, M. Chli, and R. Y. Siegwart, "Brisk: Binary robust invariant scalable keypoints," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2548–2555.

[117] A. Alahi, R. Ortiz, and P. Vandergheynst, "Freak: Fast retina keypoint," in *Computer vision and pattern recognition (CVPR), 2012 IEEE conference on*. Ieee, 2012, pp. 510–517.

[118] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," *Computer vision–ECCV 2006*, pp. 404–417, 2006.

[119] J. Heinly, E. Dunn, and J. Frahm, "Comparative evaluation of binary features," in *Computer Vision - ECCV 2012 - 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part II*, 2012, pp. 759–773.

[120] M. Heikkila and M. Pietikainen, "A texture-based method for modeling the background and detecting moving objects," *IEEE transactions on pattern analysis and machine intelligence*, vol. 28, no. 4, pp. 657–662, 2006.

[121] X. Tan and B. Triggs, "Enhanced local texture feature sets for face recognition under difficult lighting conditions," in *AMFG*, 2007, pp. 168–182.

[122] S. Liao, G. Zhao, V. Kellokumpu, M. Pietikäinen, and S. Z. Li, "Modeling pixel process with scale invariant local patterns for background subtraction in complex scenes," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 1301–1306.

[123] T. Ahonen and M. Pietikäinen, "Soft histograms for local binary patterns," in *Proceedings of the Finnish signal processing symposium, FINSIG*, vol. 5, no. 9, 2007, p. 1.

[124] J. Ren, X. Jiang, and J. Yuan, "Noise-resistant local binary pattern with an embedded error-correction mechanism," *IEEE Transactions on Image Processing*, vol. 22, no. 10, pp. 4049–4060, 2013.

[125] Y. Huang, Y. Wang, and T. Tan, "Combining statistics of geometrical and correlative features for 3d face recognition." in *BMVC*. Edinburgh, 2006, pp. 879–888.

[126] L. Nanni, S. Brahnam, and A. Lumini, "A local approach based on a local binary patterns variant texture descriptor for classifying pain states," *Expert Systems with Applications*, vol. 37, no. 12, pp. 7888–7894, 2010.

[127] O. Lahdenoja, M. Laiho, and A. Paasio, "Reducing the feature vector length in local binary pattern based face recognition," in *Image Processing, 2005. ICIP 2005. IEEE International Conference on*, vol. 2. IEEE, 2005, pp. II–914.

[128] D. T. Nguyen, P. O. Ogunbona, and W. Li, "A novel shape-based non-redundant local binary pattern descriptor for object detection," *Pattern Recognition*, vol. 46, no. 5, pp. 1485–1500, 2013.

[129] Y. Mu, S. Yan, Y. Liu, T. Huang, and B. Zhou, "Discriminative local binary patterns for human detection in personal album," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on.* IEEE, 2008, pp. 1–8.

[130] D. T. Nguyen, Z. Zong, P. Ogunbona, and W. Li, "Object detection using non-redundant local binary patterns," in *Image Processing (ICIP), 2010 17th IEEE International Conference on.* IEEE, 2010, pp. 4609–4612.

[131] A. Fathi and A. R. Naghsh-Nilchi, "Noise tolerant local binary pattern operator for efficient texture analysis," *Pattern Recognition Letters*, vol. 33, no. 9, pp. 1093–1100, 2012.

[132] L. Liu, P. W. Fieguth, D. A. Clausi, and G. Kuang, "Sorted random projections for robust rotation-invariant texture classification," *Pattern Recognition*, vol. 45, no. 6, pp. 2405–2418, 2012.

[133] Y. Zhao, D.-S. Huang, and W. Jia, "Completed local binary count for rotation invariant texture classification," *IEEE Transactions on Image Processing*, vol. 21, no. 10, pp. 4492–4497, 2012.

[134] J. Ryu, S. Hong, and H. S. Yang, "Sorted consecutive local binary pattern for texture classification," *IEEE Transactions on Image Processing*, vol. 24, no. 7, pp. 2254–2265, 2015.

[135] X. Qi, R. Xiao, C.-G. Li, Y. Qiao, J. Guo, and X. Tang, "Pairwise rotation invariant co-occurrence local binary pattern," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 36, no. 11, pp. 2199–2213, 2014.

[136] X. Qi, Y. Qiao, C. Li, and J. Guo, "Multi-scale joint encoding of local binary patterns for texture and material classification," in *British Machine Vision Conference, BMVC 2013, Bristol, UK, September 9-13, 2013*, 2013.

[137] E. González, A. Fernández, and F. Bianconi, "General framework for rotation invariant texture classification through co-occurrence of patterns," *Journal of mathematical imaging and vision*, vol. 50, no. 3, pp. 300–313, 2014.

[138] C. Shan and T. Gritti, "Learning discriminative lbp-histogram bins for facial expression recognition." in *BMVC*, 2008, pp. 1–10.

[139] J. Zhang, J. Liang, and H. Zhao, "Local energy pattern for texture classification using self-adaptive quantization thresholds," *IEEE transactions on image processing*, vol. 22, no. 1, pp. 31–42, 2013.

[140] J. Kannala and E. Rahtu, "Bsif: Binarized statistical image features," in *Pattern Recognition (ICPR), 2012 21st International Conference on*. IEEE, 2012, pp. 1363–1366.

[141] Z. Li, G. Liu, Y. Yang, and J. You, "Scale-and rotation-invariant local binary pattern using scale-adaptive texton and subuniform-based circular shift," *IEEE Transactions on Image Processing*, vol. 21, no. 4, pp. 2130–2140, 2012.

[142] R. Davarzani, S. Mozaffari, and K. Yaghmaie, "Scale-and rotation-invariant texture description with improved local binary pattern features," *Signal Processing*, vol. 111, pp. 274–293, 2015.

[143] L. Liu, B. Yang, P. Fieguth, Z. Yang, and Y. Wei, "Brint: A binary rotation invariant and noise tolerant texture descriptor," in *International Conference on Image Processing*, Melbourne, 2013.

[144] T. Ahonen, J. Matas, C. He, and M. Pietikäinen, "Rotation invariant image description with local binary pattern histogram fourier features," *Image analysis*, pp. 61–70, 2009.

[145] A. Satpathy, X. Jiang, and H.-L. Eng, "Lbp-based edge-texture features for object recognition," *IEEE Transactions on Image Processing*, vol. 23, no. 5, pp. 1953–1964, 2014.

[146] X. Wang, T. X. Han, and S. Yan, "An hog-lbp human detector with partial occlusion handling," in *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE, 2009, pp. 32–39.

[147] S. U. Hussain and W. Triggs, "Feature sets and dimensionality reduction for visual object detection," in *BMVC 2010-British Machine Vision Conference*. BMVA Press, 2010, pp. 112–1.

[148] B. F. Klare and A. K. Jain, "Heterogeneous face recognition using kernel prototype similarities," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 6, pp. 1410–1422, 2013.

[149] C. H. Chan, M. A. Tahir, J. Kittler, and M. Pietikäinen, "Multiscale local phase quantization for robust component-based face recognition using kernel fusion of multiple descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 5, pp. 1164–1177, 2013.

[150] V. Ojansivu and J. Heikkilä, "Blur insensitive texture classification using local phase quantization," in *International conference on image and signal processing*. Springer, 2008, pp. 236–243.

[151] J. Chen, S. Shan, C. He, G. Zhao, M. Pietikäinen, X. Chen, and W. Gao, "WLD: A robust local image descriptor," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1705–1720, 2010.

[152] G. Sharma, S. ul Hussain, and F. Jurie, "Local higher-order statistics (lhs) for texture categorization and facial analysis," *Computer Vision–ECCV 2012*, pp. 1–12, 2012.

[153] R. Maani, S. Kalra, and Y.-H. Yang, "Noise robust rotation invariant features for texture classification," *Pattern Recognition*, vol. 46, no. 8, pp. 2103–2116, 2013.

[154] Z. Lei, M. Pietikäinen, and S. Z. Li, "Learning discriminant face descriptor," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 2, pp. 289–302, 2014.

[155] H. Lategahn, S. Gross, T. Stehle, and T. Aach, "Texture classification by modeling joint distributions of local patterns with gaussian mixtures," *IEEE Transactions on Image Processing*, vol. 19, no. 6, pp. 1548–1557, 2010.

[156] T. Ahonen and M. Pietikäinen, "Image description using joint distribution of filter bank responses," *Pattern Recognition Letters*, vol. 30, no. 4, pp. 368–376, 2009.

[157] J. Sánchez, F. Perronnin, T. Mensink, and J. Verbeek, "Image classification with the fisher vector: Theory and practice," *International journal of computer vision*, vol. 105, no. 3, pp. 222–245, 2013.

[158] R. Maani, S. Kalra, and Y.-H. Yang, "Robust edge aware descriptor for image matching," in *Asian Conference on Computer Vision*. Springer, 2014, pp. 553–568.

[159] L. Liu and P. Fieguth, "Texture classification from random features," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 3, pp. 574–586, 2012.

[160] S. Mallat, "Group Invariant Scattering," *Communications on Pure and Applied Mathematics*, vol. 65, no. 10, pp. 1331–1398, 2012.

[161] L. Sifre and S. Mallat, "Rigid-motion scattering for texture classification," *CoRR*, vol. abs/1403.1687, 2014.

[162] S. Mallat, *A wavelet tour of signal processing: the sparse way*. Academic press, 2008.

[163] Y. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, "Efficient backprop," in *Neural networks: Tricks of the trade*. Springer, 1998, pp. 9–50.

[164] Y. LeCun, K. Kavukcuoglu, and C. Farabet, "Convolutional networks and applications in vision," in *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*. IEEE, 2010, pp. 253–256.

[165] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[166] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European conference on computer vision*. Springer, 2014, pp. 818–833.

[167] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal, The*, vol. 27, no. 3, pp. 379–423, Jul. 1948.

[168] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, T. M. Cover and J. A. Thomas, Eds. John Wiley & Sons, 1991.

[169] M. K. Nguyen and A. M. Djafari, "Bayesian approach with the maximum entropy principle in image reconstruction from microwave scattered field data," *IEEE Trans. Medical Imaging*, vol. 13, pp. 254–262, 1994.

[170] Y. He, A. B. Hamza, and H. Krim, "A generalized divergence measure for robust image registration," *IEEE Trans. Signal Processing*, vol. 51, no. 5, pp. 1211–1220, May 2003.

[171] W. Ni, N.-S. Vu, and A. Caplier, "Lucas-kanade based entropy congealing for joint face alignment," *Image Vision Comput.*, vol. 30, no. 12, pp. 954–965, 2012.

[172] N.-S. Vu and A. Caplier, "Illumination-robust face recognition using the retina modelling," in *ICIP*. IEEE, 2009, pp. 3289–3292.

[173] B. Caputo, E. Hayman, M. Fritz, and J.-O. Eklundh, "Classifying materials in the real world," *Image Vision Comput.*, vol. 28, no. 1, pp. 150–163, 2010.

[174] M. Varma and A. Zisserman, "A statistical approach to material classification using image patch exemplars," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 11, pp. 2032–2047, 2009.

[175] S. Liao, M. W. K. Law, and A. C. S. Chung, "Dominant local binary patterns for texture classification," *IEEE Transactions on Image Processing*, vol. 18, no. 5, pp. 1107–1118, May 2009.

[176] M. Varma and A. Zisserman, "A statistical approach to texture classification from single images," *Int. Journal of Computer Vision*, vol. 62, no. 1-2, pp. 61–81, 2005.

[177] Z. Guo, X. Wang, J. Zhou, and J. You, "Robust texture image representation by scale selective local binary patterns," *IEEE Transactions on Image Processing*, vol. 25, no. 2, pp. 687–699, 2016.

[178] J. Chen, S. Shan, C. He, G. Zhao, M. Pietikäinen, X. Chen, and W. Gao, "Wld: A robust local image descriptor," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1705–1720, 2010.

[179] M. Cimpoi, S. Maji, I. Kokkinos, and A. Vedaldi, "Deep filter banks for texture recognition, description, and segmentation," *International Journal of Computer Vision*, vol. 118, no. 1, pp. 65–94, 2016.

[180] V.-L. Nguyen, N.-S. Vu, and P.-H. Gosselin, "A scattering transform combination with local binary pattern for texture classification," *14th International Workshop on Content-based Multimedia Indexing*, 2016.

[181] V.-L. Nguyen, N.-S. Vu, H.-H. Phan, and P.-H. Gosselin, "An integrated descriptor for texture classification," *23rd International Conference on Pattern Recognition*, 2016.

[182] ——, "Lbp-and-scatnet-based combined features for efficient texture classification," *Multimedia Tools and Applications*, vol. 76, no. 21, pp. 22 425–22 444, 2017.

[183] A. Vedaldi and B. Fulkerson, "VLFeat: An open and portable library of computer vision algorithms," `http://www.vlfeat.org/`, 2010.

[184] A. Vedaldi and K. Lenc, "Matconvnet – convolutional neural networks for matlab," in *Proceeding of the ACM Int. Conf. on Multimedia*, 2015.

[185] X. Hong, G. Zhao, M. Pietikäinen, and X. Chen, "Combining LBP difference and feature correlation for texture description," *IEEE Transactions on Image Processing*, vol. 23, no. 6, pp. 2557–2568, 2014.

[186] J. Ren, X. Jiang, and J. Yuan, "Noise-resistant local binary pattern with an embedded error-correction mechanism," *IEEE Transactions on Image Processing*, vol. 22, no. 10, pp. 4049–4060, 2013.

[187] A. Hafiane, G. Seetharaman, and B. Zavidovique, "Median binary pattern for textures classification," in *Image Analysis and Recognition, 4th International Conference, ICIAR 2007, Montreal, Canada, August 22-24, 2007, Proceedings*, 2007, pp. 387–398.

[188] R. E. Broadhurst, "Statistical estimation of histogram variation for texture classification," in *Proc. Intl. Workshop on texture analysis and synthesis*, 2005, pp. 25–30.

[189] M. Crosier and L. D. Griffin, "Using basic image features for texture classification," *International Journal of Computer Vision*, vol. 88, no. 3, pp. 447–460, 2010.

[190] E. Hayman, B. Caputo, M. Fritz, and J. Eklundh, "On the significance of real-world conditions for material classification," in *European Conference on Computer Vision*, 2004.

[191] Y. Xu, X. Yang, H. Ling, and H. Ji, "A new texture descriptor using multifractal analysis in multi-orientation wavelet pyramid," in *CVPR*, 2010, pp. 161–168.

[192] Y. Xu, S. Huang, H. Ji, and C. Fermüller, "Combining powerful local and global statistics for texture description," in *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, 2009, pp. 573–580.

[193] Y. Xu, H. Ji, and C. Fermüller, "Viewpoint invariant texture description using fractal analysis," *International Journal of Computer Vision*, vol. 83, no. 1, pp. 85–100, 2009.

[194] V.-L. Nguyen, N.-S. Vu, and P.-H. Gosselin, "A handcrafted normalized-convolution network for texture classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1238–1245.

[195] G. Farnebäck, "Polynomial expansion for orientation and motion estimation," Ph.D. dissertation, Linköping University Electronic Press, 2002.

[196] F. Perronnin and C. Dance, "Fisher kernels on visual vocabularies for image categorization," in *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on.* IEEE, 2007, pp. 1–8.

[197] G. Schaefer and N. P. Doshi, "Multi-dimensional local binary pattern descriptors for improved texture analysis," in *Pattern Recognition (ICPR), 2012 21st International Conference on.* IEEE, 2012, pp. 2500–2503.

[198] G. Zhao, T. Ahonen, J. Matas, and M. Pietikainen, "Rotation-invariant image and video description with local binary pattern features," *IEEE Transactions on Image Processing*, vol. 21, no. 4, pp. 1465–1477, 2012.

[199] H. Sun, D. Gao, Z. Zhao, and X. Tang, "An approach to in-process surface texture condition monitoring," *Robotics and Computer-Integrated Manufacturing*, vol. 48, pp. 254–262, 2017.

[200] R. Ren, T. Hung, and K. C. Tan, "A generic deep-learning-based approach for automated surface inspection," *IEEE transactions on cybernetics*, vol. 48, no. 3, pp. 929–940, 2018.

[201] I. Chingovska, A. Anjos, and S. Marcel, "On the effectiveness of local binary patterns in face anti-spoofing," in *Biometrics Special Interest Group (BIOSIG), 2012 BIOSIG-Proceedings of the International Conference of the.* IEEE, 2012, pp. 1–7.

[202] F. Juefei-Xu, V. N. Boddeti, and M. Savvides, "Local binary convolutional neural networks," in *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, vol. 1, 2017.