LOCAL AND DEEP TEXTURE FEATURES FOR CLASSIFICATION OF NATURAL AND BIOMEDICAL IMAGES

A Thesis presented to

the Faculty of the Graduate School

at the University of Missouri

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

---

by

ZAKARIYA AHMED ORAIBI

Prof. KANNAPPAN PALANIAPPAN, Thesis Supervisor

July 2019

The undersigned, appointed by the Dean of the Graduate School, have examined the dissertation entitled:

LOCAL AND DEEP TEXTURE FEATURES FOR CLASSIFICATION OF NATURAL AND BIOMEDICAL IMAGES

presented by ZAKARIYA AHMED ORAIBI,
a candidate for the degree of Doctor of Philosophy and hereby certify that, in their opinion, it is worthy of acceptance.

Prof. Kannappan Palaniappan

Prof. David R. Larsen

Prof. Jianlin Cheng

Dr. Filiz Bunyak

ACKNOWLEDGMENTS

I would like to express my gratitude to my government in Iraq for sponsoring my scholarship and for providing this precious opportunity for me to study abroad in a prestigious university in the United States. In particular, I would like to thank all members of Higher Committee for Education Development (HCED) in Iraq for being responsible to provide me this scholarship and for following up with me whenever I needed them. I would like also to thank my wife and my kids for joining and supporting me throughout this long journey.

My deepest gratitude also extends to my adviser professor Kannappan Palaniappan for being a brilliant mentor and a big source of inspiration in my PhD research especially for the publications we authored together. Besides Dr. Palaniappan, I would like to thank my dissertation committee members: Professor David Larsen, Professor Jianlin Cheng, and Dr. Filiz Bunyak, not only for their observant comments and encouragement, but also for the questions and advises that allowed me to widen my research from various perspectives. I would like to thank professor Prasad Calyam and Dr. Dmitrii Chemodanov with whom I started my first publications in my PhD journey. In addition, I would like also to thank my lab mates and everyone in our department who provided me with precious advise through seminars and helped me to make meet deadlines.

I would like to thank my friends in the city of Columbia, MO for motivating me to continue and finish my scholarship program. I would like also to thank all my family members and friends in Basrah, Iraq for pushing me to finish the program and for helping me whenever I needed anything at anytime.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Developing efficient feature descriptors is very important in many computer vision applications including biomedical image analysis. In the past two decades and before the popularity of deep learning approaches in image classification, texture features proved to be very effective to capture the gradient variation in the image. Following the success of the Local Binary Pattern (LBP) descriptor, many variations of this descriptor were introduced to further improve the ability of obtaining good classification results. However, the problem of image classification gets more complicated when the number of images increases as well as the number of classes. In this case, more robust approaches must be used to address this problem.

In this thesis, we address the problem of analyzing biomedical images by using a combination of local and deep features. First, we propose a novel descriptor that is based on the motif Peano scan concept called Joint Motif Labels (JML). After that, we combine the features extracted from the JML descriptor with two other descriptors called Rotation Invariant Co-occurrence among Local Binary Patterns (RIC-LBP) and Joint Adaptive Medina Binary Patterns (JAMBP). In addition, we construct another descriptor called Motif Patterns encoded by RIC-LBP and use it in our classification framework. We enrich the performance of our framework by combining these local descriptors with features extracted from a pre-trained deep network called VGG-19. Hence, the 4096 features of the Fully Connected 'fc7' layer are extracted and combined with the proposed local descriptors. Finally, we show that Random Forests (RF) classifier can be used to obtain superior performance in the field of biomedical image analysis. Testing was performed on two standard biomedical datasets and another three standard texture datasets. Results show that our framework can beat state-of-the-art accuracy on the biomedical image analysis and the combination of local features produce promising results on the standard texture datasets.

Chapter 1

Introduction

1.1 Texture definition and properties

Texture can be defined as a measure of the variation of a surface, shape, shadows, absorption, and illumination of something. The visual appearance of an object may consist of elements like sand or marble which give a robust definition to texture. Another important factor that defines texture is the distance human beings use to view texture regions. Different interpretations for texture regions can be obtained at different distance degrees when perceived by our visual system. For example, if we fix all the parameters mentioned previously that define texture like shape and illumination and change the camera position to capture that texture region, we will obtain different resultant image. This variability in image appearance of a texture region makes it difficult in computer vision related problems like image classification.

It has been difficult to give a formal mathematical representation for texture analysis problem. However, two approaches were used to tackle this issue in computer vision: structured approach and statistical approach. Haralick et al. [5] divided the input image into blocks and considered the statistical nature of texture by calculating

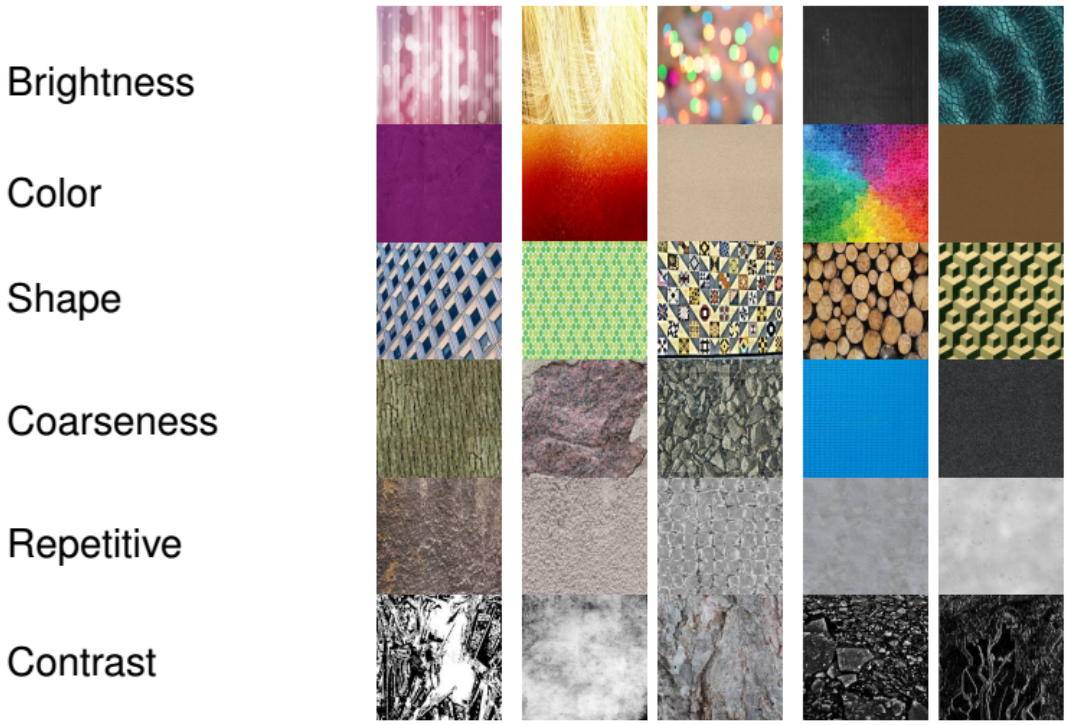


Figure 1.1: Properties of Texture in Image Analysis: fineness, smoothness, roughness, granulation, randomness, periodic, lineation, mottled, irregular, hummocky. [Haralick 1992]

the probability distribution of a given block to extract 14 textural features. These features represent texture characteristics like contrast and homogeneity as shown if Figure 1.1. Tamura et al. [6] extracted six textural features similar to those visually perceived by human visual system including coarseness, line-likeness, and regularity. On the other hand, Rao et al. [7] extracted three high level texture features using orthogonal dimensions as follows: repetitive vs. non-repetitive, non-directional with high contrast vs. directional with low contrast, and simple granular textures vs. fine-grained complex textures. Recently, Cimpoi et al. [8] introduced 47 selected attributes that capture a wide range of visual properties of texture and introduced a new describable texture dataset. They used a combination of global and deep features in benchmarking texture challenging texture datasets. The techniques used in this work represent a new state-of-the-art due to the powerful representation of texture features using deep and global features.

In general, the texture classification problem involves two stages: feature extraction and classification. The majority of research is conducted on the first stage where designing robust, efficient, and discriminative features is very important to ensure a better classification accuracy. As a result, thorough research has been made to extract such features with surveys include the work of Zhang et al. [9], Pietikainen et al. [10], and more recently the work of Liu et al. [11] which evaluated the performance of state-of-the-art LBP descriptors and deep learning based descriptors on different texture benchmarks. For the classification stage, most local descriptors use the k-NN classifier. However, for deep learning based descriptors, SVM classifier also proved to be very robust and generated high results.

1.2 Motivation

Analyzing images based on texture features can benefit many applications related to computer vision field including remote sensing, inspecting material and industrial surface, texture synthesis, content-based image retrieval and biomedical image classification and segmentation. Other features, like color, might not be applicable to the desired application. Moreover, color features are more sensitive to illumination changes. On the other hand, texture features can be designed to overcome this issue and other issues related to rotation and translation invariance. For decades, texture analysis has been an active area of research. Many descriptors have been proposed, however, improvements have always been slow as the problem gets more complicated with large scale images that contain different types of texture patterns. In addition, Many factors in the design of texture descriptors should be considered in order to be useful and efficient to suit the desired application. These factors include the size of the feature vector resulted from applying the descriptor in order to facilitate the classification process and the speed of feature extraction which should be fast enough to cope with the increase number of images in the given database. Given that many local descriptors have been designed to extract texture features, new researches are working to perform texture analysis using a combination of these descriptors.

Local Binary Pattern (LBP) is considered as one robust and efficient texture descriptors that has been used widely in many applications in the past two decades. The descriptor was first introduced in 1996 by Ojala et al. [12] following the work of Harwood et al. [13] that is considered the base stone of LBP operator. LBP works on gray-scale images and its strength comes from its ability to handle the rotation element that can occur to texture patterns. Moreover, LBP is scalable and can be extended using multi-scale analysis or to be joined by additional information from the given image. Another local descriptor that was used successfully in content based image retrieval is called Motif Cooccurrence Matrix (MCM) by Jhanwar et al. [1].

This descriptor is based on space filling curves work [14, 15] and it is very efficient in terms of computation speed and storage.

It is worth to mention that in 1990s statistical distribution of texture features in the image region dominated the research in texture classification field. Representing texture areas using statistic properties of the local textures led to the emergence of "textons" with models such as Bag-of-Words (BoW) and Fisher Vector (FV) dominating this area of research [16, 17, 18]. Other texture representation methods include Gabor wavelets [19], Leung and Malik (LM) filters [20], MR8 filters [18], SIFT descriptor [9], and Patch descriptors [21]. However, using these statistical methods is not as easy as using raw pixel features descriptor like LBP. Creating the vocabulary "textons" for these methods leads to consume more time that can be avoided by using LBP. Moreover, LBP has other characteristics like its simplicity in terms of implementation and use, flexibility, and its ability to handle rotation invariance in the image efficiently.

Recently, deep learning methods showed superior performance in representing texture for image classification and segmentation [22]. A key factor to the success of CNN is the ability to handle large labeled datasets. The use of GPU made it easy for CNN methods to generate and store millions of parameters (weights) that require a very large memory in a considerable amount of time. In addition, researches showed that CNN features extracted from pre-trained networks are able to transfer to other different problems, including texture classification [23, 24, 25, 26, 27]. In general, pre-trained and finetuned CNN models are the techniques used in texture classification since they had a great influence in image understanding.

1.3 Contributions

Following the success of LBP descriptor two decades ago, new robust extensions of the basic LBP have been designed by exploiting many information from the image and the relationship of the surrounding pixels. As a result, it is possible to exploit these variations of LBP and combine them with other local descriptors like extensions of MCM descriptor to create a robust framework for texture classification. In addition, the new trend for image classification is to use deep learning techniques. Several architectures were proposed recently and showed superior performance. However, for the texture recognition problem, the successful approaches used transfer learning by extracting the features from a pre-trained network like VGG-19 and use a classifier like SVM to classify images and produce the final accuracy.

In this thesis, we propose to perform image classification using a combination of multiple local descriptors and CNN features. First, a new descriptor is developed based on the motif Peano scan concept. We extract twelve motif patterns from each 2 x 2 neighborhood of the original image. After that, three motif-labels are found based on these motifs (Min, Med, and Max). Then each matrix is joined with additional information extracted from the input image which are mean and variance to produce a new 3D joint moment. Finally, all three moments are combined together to form the final descriptor. The motif patterns generated from this process was further exploited to create a second descriptor by encoding these patterns with one of the robust LBP variations. Second, two robust extensions of LBP descriptor are used in a late fusion mechanism along with the newly designed descriptors. Finally, a classifier is used based on these local descriptors to perform classification and produce the result.

The framework of local descriptors proposed in this thesis was tested extensively on five different datasets. Images from these databases represent bimodical and nature images, the latter are used specifically to test the robustness of texture descriptors. Biomedical image analysis was performed on two databases called Human Epithelial

Type-2 (HEp-2) cell and specimen classification. The other three texture databases are: KTH-TIPS-2a,b and DTD. The recently introduced DTD dataset represents a big challenge with many classes and many images in each class which makes the use of a single texture operator not sufficient to produce high classification accuracy. The additional CNN features to the framework was only tested on both biomedical image analysis and standard texture databases, however, we obtained superior peformance with biomedical images while the performance is still developing on texture datasets.

In the classification stage, we have used three classifiers: SVM, k-NN, and Random Forests (RF). We performed experiments with different kernels using SVM classifier and with different number of trees using RF classifier. Our results indicated that using a combination of local descriptors, we can generate better accuracy than using a single local descriptor. In addition, combining our proposed features with deep features extracted from the Fully Connected layer can produce superior result especially on HEp-2 specimen dataset. We demonstrate a comparison between our results and previously obtained results using different techniques for feature extraction and classification.

In addition to the main contribution in the machine learning field, I also participated in two projects related to visual cloud computing and networking. These projects where administrated by Professor Prasad Calyam and Dr. Dmitrii Chemodanov and where published in top journals. Here is a description of these projects:

Edge Routing and IoT [28] (supported by NSF, Coulter Foundation, RFBR, Army Research Lab) Applications that cater to the needs of disaster incident response generate large amount of data and demand large computational resource access. Such datasets are usually collected in real-time at the incident scenes using different Internet of Things (IoT) devices. Hierarchical clouds, i.e., core and edge clouds, can help these applications' real-time data orchestration challenges as well as with their IoT operations scalability, reliability and stability by overcoming infrastructure limi-

tations at the ad-hoc wireless network edge. Edge routing is a crucial infrastructure management orchestration mechanism for such systems. However, current Edge geographic routing (or greedy forwarding) approaches designed for early wireless ad-hoc networks lack efficient solutions for disaster incident-supporting applications, given the high-speed and low-latency data delivery that edge cloud gateways impose. In this set of activities, we propose a novel Artificial Intelligent (AI)-augmented geographic routing approach (AGRA), that relies on an area knowledge obtained from the satellite imagery (available at the edge cloud) by applying deep learning. In particular, we propose a stateless greedy forwarding algorithm that uses such an environment learning to proactively avoid the local minimum problem by diverting traffic with an algorithm that emulates electrostatic repulsive forces. We have shown that our Greedy Forwarding achieves in the worst case a 3.291 path stretch approximation bound with respect to the shortest path (without assuming presence of symmetrical links or unit disk graphs), and thus, improves the application level throughput under severe node failures and high mobility challenges of disaster response scenarios. Initial results from this study was published in Elsevier FGCS and have been further extended by the master student (who I mentored) by proposing a policy-based version of AGRA that trade-offs energy and throughput for making offloading decisions (i.e., to an edge cloud or a core cloud) of the actual face recognition application.

Visual Cloud Computing for Incident-Supporting Situation Awareness [29]: In the event of natural or man-made disasters, geospatial video analytics is valuable to provide situational awareness that can be extremely helpful for first responders. However, geospatial video analytics demands massive imagery/video data 'collection' from Internetof-Things (IoT) and their seamless 'computation/consumption' within a geo-distributed (edge/core) cloud infrastructure in order to cater to user Quality of Experience (QoE) expectations. Thus, the edge computing needs to be designed with a reliable performance while interfacing with the core cloud to run computer

vision algorithms. This is because infrastructure edges near locations generating imagery/video content are rarely equipped with high-performance computation capabilities

In this area of research, we address challenges of interfacing edge and core cloud computing within the geo-distributed infrastructure as a novel 'function-centric computing' paradigm that brings new insights to computer vision, edge routing and network virtualization areas. Specifically, we propose our new/improved solution approaches based on function-centric computing for the two problems of: (i) high-throughput data collection from IoT devices at the wireless edge, and (ii) seamless data computation/consumption within the geo-distributed (edge/core) cloud infrastructure. To address (i), we present a novel deep learning-augmented geographic edge routing that relies on physical area knowledge obtained from satellite imagery.

Chapter 2

Literature Review

2.1 Local Binary Pattern Descriptor

The LBP descriptor is considered as one of the most powerful local descriptors. The descriptor was first mentioned in [30]. Later on, the developed LBP descriptor introduced in 2002 by Ojala et al. [31] focuses on image classification using gray-scale images. The strength of LBP comes from its ability to encode a unique local binary texture patterns called uniform patterns and the ability of designing rotation invariant local features. These uniform patterns are essential since they represent the majority of texture features in a specific neighborhood. In addition, this descriptor has the ability to detect texture features at any resolution. Hence, it is also possible to enhance the classification performance using different image resolutions to extract LBP features. The idea of extracting features using multi-resolution analysis assumes that the features extracted from each resolution are independent. Then, we can combine these descriptors in order to get higher classification accuracy. Another advantage of using LBP is the simplicity of this descriptor which makes it computationally inexpensive.

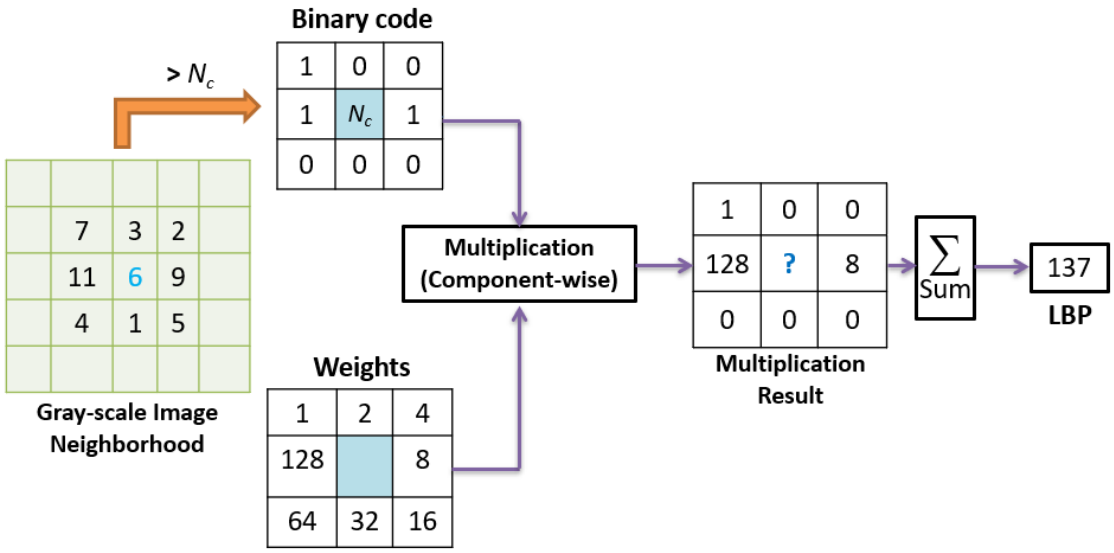


Figure 2.1: Mechanism of LBP using a 3 x 3 image neighborhood. The original image must be a gray-scale. Each center pixel of the 3×3 neighborhood is compared to the surrounding pixels. Then, if the value of the neighborhood is greater than the center pixel, it will be replaced by 1, otherwise, it will be replaced by 0. After that, all neighborhood values after comparison are multiplied by the corresponding weight values. Finally, the LBP value of the center pixel will be the summation of all weights.

The LBP operator works on a small neighborhood of the image, ex. 3 x 3. Two steps are involved in the computation of LBP descriptor. The first step is to extract the binary patterns from the circular neighborhood. The second step is to compute the histogram distribution of these patterns. In the first step, the center pixel is used to threshold the surrounding pixels in a small image patch as shown in Figure 2.1. To put this in equations, assume N_c is the center pixel of an image, N_p is the surrounding pixels, then:

$$LBP_{p,r}(N_c) = \sum_{p=0}^{p-1} g(N_p - N_c)2^p \tag{2.1}$$

$$g(x) = \begin{cases} 0, \text{if x} < 0 \\ 1, \text{if x} \geq 0 \end{cases}$$

where, p is the sampling points and r is the radius from the center pixel. $g(x)$ is the binary thresholding function.

After obtaining the signs of each LBP pattern from this thresholding process, weights are given to each pattern as illustrated in Figure 2.1. The final LBP for the given center pixel is the summation of all weights. In the implementation, $LBP_{P,R}$ descriptor requires interpolation to obtain the diagonal values.

After computing the LBP for each circular neighborhood, rotation invariance is necessary to ensure that any rotation to the input image will not affect the final binary patterns. Early attempts on introducing rotation invariant features include generalized cooccurrence matrices by Davis et al. [32] and texture anisotropy by Chetverikov et al. [33]. Many early methods relied on transforming a well defined noninvariant approach to a successful invariant approach. For example, the Circular Simultaneous Autoregressive (CSAR) technique introduced by Kashyap and Khotanzad [34], the Multiresolution Simultaneous Autoregressive (MRSAR) model by Mao and Jain [35], and the works of Wu and Wei [36] and Cohen et al. [37]. For filtering based approaches like Gabor wavelets, two approaches were used in this case: either the input image is filtered first and invariant features are calculated or the variant features are converted into rotation invariant features [38, 39, 40, 41, 42, 43]. Other methods joined invariance by making use of both spatial scale and rotation [44, 37, 42, 45, 46]. In [47] Wang and Healey were the first to introduce rotation invariance with respect to three main image properties: spatial scale, rotation, and grey scale.

The LBP operator introduced in 2002 by Ojala et al. [31] focuses on grey-scale and rotation invariant texture classification. A computationally simple approach was introduced which is robust to grey-scale variations and can handle a wide range of rotated textures efficiently. This approach is based on the local binary patterns. Since $\text{LBP}_{P,R}$ produces 2^P output values, when the image is rotated at any angle, the grey values will move along the perimeter of the circle around the given LBP. In this case,

rotating a LBP will result in different LBP values. The only patterns that will not be affected by rotation are the ones that are either have all 0s or 1s binary patterns. To eliminate this effect, a unique identifier is assigned to each rotation invariant LBP in order to perform a circular right shift depending on the P-bit a specific number of times. In the case of P = 8, that is, $LBP^{ri}_{8,R}$ can have 36 different values. During experience, it was shown that achieving only rotation invariance is not enough to discriminate texture features in the given image. However, certain local binary patterns are able to provide such discrimination and they represent the majority of features in a given 3 x 3 neighborhood patch. These patterns are called "uniform" and contain only very few spatial transition (between 0/1). $LBP^{riu2}_{8,R}$ operator means that the rotation invariant used is of type "uniform" with at most 2 transitions among the circular patterns. Finally, the histogram of the pattern labels is calculated and used in texture analysis.

2.1.1 Variations of LBP Descriptor

Following the success of the LBP descriptor in recognizing texture features. Many variations of the original operator were introduced to further improve the classification performance of the original descriptor. Since the basic form of LBP includes only calculating the difference between the center pixel and the surrounding pixels and then computing the histogram of the resultant patterns, additional information can be used and joined or combined with the LBP descriptor to generate a more robust operator especially for difficult texture recognition problems. In addition, many researches found it easy to build on the basic LBP descriptor considering its simplicity in terms of implementation, its ability to handle illumination and rotation changes of the given images, and the advantage it has over other methods like Bag of Visual Words (BoVW) where no dictionary is required to use the LBP model. We will discuss the most important and well known variations since our framework of texture

classification uses some of these state-of-the-art variations.

Completed Local Binary Pattern (CLBP): CLBP descriptor is considered one of the powerful variations of the original LBP operator. It was introduced by Guo et al. [48] in 2010 and many other local operators that followed, used the mechanism introduced by CLBP. The CLBP descriptor starts by creating three local descriptors: CLBP_Sign (CLBP_S), CLBP_Magnitude (CLBP_M), and CLBP_Center (CLBP_C). CLBP_S and CLBP_M simply are derived by calculating the absolute difference between the center pixel and the surrounding pixels in a given neighborhood like 3 x 3. This descriptor is more efficient and robust to illumination changes since the original LBP uses only the sign vector not incorporating the magnitude. CLBP_C uses a global thresholding for all the grey pixels of the original input image by comparing them to the mean pixel values of the entire image. A CLBP framework is then created by combining these three local descriptors to form the final feature map. Two mechanisms were used to perform this combination: joint combination or concatenation. The first method is similar to 2-D joint histogram, instead, we can create a 3-D joint histogram of the three descriptors and the final framework will be denoted by "CLBP_S/M/C". The second way is by combining two forms of these descriptors, like "CLBP_S/C" or "CLBP_M/C" and the final histogram is concatenated with "CLBP_M" or "CLBP_S" to generate "CLBP_M_S/C" or "CLBP_S_M/C". After generating the final histogram, a classifier is used like nearest neighbor (NN) to perform image classification based on the CLBP extracted features.

Completed Local Binary Count (CLBC): CLBC descriptor was proposed in 2012 by Zhao et al. [49]. The idea behind composing CLBP descriptor is similar to CLBP framework descriptor. However, CLBP depends on LBP basic descriptor while CLBC depends on a novel descriptor called Local Binary Count (LBC). In LBC, instead of converting each image pixel into a binary pattern, LBC only counts the number of 1's in the binary neighborhood after thresholding the center pixel

and the surrounding pixels of the given image patch. Finally, similar to CLBP, a CLBC framework is formulated using the three proposed descriptors as in CLBP: CLBC-Sign, CLBC-Magnitude, CLBC-Center and join them using the two methods proposed in CLBP descriptor. The advantage of CLBC is that it is computationally less expensive in the classification process. In addition, CLBC classification performance is shown to be slightly better than CLPB descriptor.

discriminative Completed Local Binary Pattern (disCLBP): In 2012, Guo et al. proposed a new learning descriptor composed of three-layered model [50]. This model is general and can be incorporated with other variations of LBP like CLBP and be used to recognize texture features. The basic idea behind this descriptor is to use an efficient feature selection model depending on the given texture classes. The three layered model convey feature robustness, discriminative power, and representation capability. The first property is achieved by learning a model with subset of features that are frequently appeared in the image. The second property is achieved by selecting the dominant patterns in each texture class in order to remove outlier patterns in each image. The third property is achieved by constructing a histogram of the union of all dominant patterns and this histogram will serve as vector representing the given image. The NN classifier is used after extracting the features to perform image classification.

Completed Local Binary Pattern Histogram Fourier Features (CLBPHF): LBP-HF descriptor was introduced in 2009 by Ahonen et al. [51]. In LBP-HF, the rotation invariant property of the extracted features is defined for the whole region to be described. Moreover, LBP-HF is a highly discriminative descriptor. Discrete Fourier Transform is used to construct such features which are computed along the input histogram rows. The NN classifier was used to perform texture and material classification along with applying this descriptor on a face recognition database.

Local Binary Pattern Variance (LBPV): In 2010, Guo et al. proposed a new

rotation invariant descriptor called LBPV [52]. Since LBP is a local operator, it has a disadvantage of losing global spatial information. In LBPV, the global variance for each local region is found after finding the LBP code of that region. Then, LBP patterns are joined with variance by finding the summation for each pattern's corresponding variance value as shown in Figure 2.2.

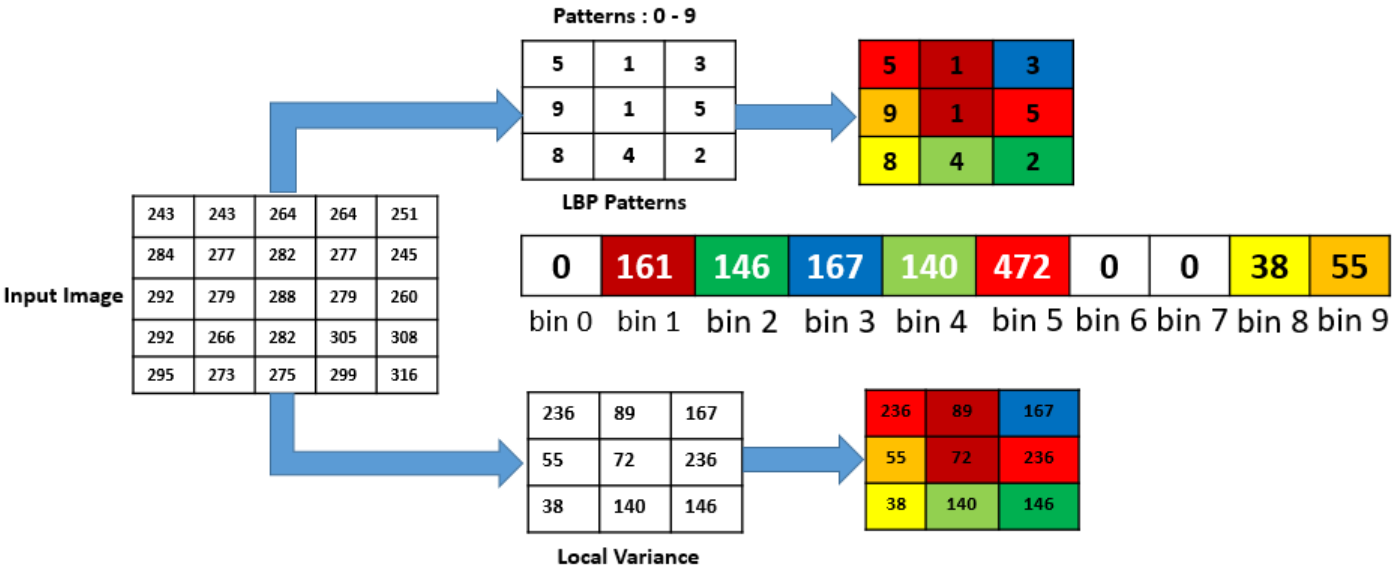


Figure 2.2: Our Illustration of the Mechanism of LBPV Descriptor. After finding the LBP patterns in the gray-scale image and the global variance for each local region, LBP patterns and the variance are joined by computing the summation of the corresponding variance for each LBP.

The advantage of this descriptor is that it assigns weights for each binary pattern. Since high frequency texture regions have higher variances, this will give LBPV higher discrimination ability over texture regions.

Local Ternary Pattern (LTP): LTP was proposed by Tan and Triggs in 2010 [53]. The main difference between LBP and LTP is that the latter is less sensitive to noise and provides more discrimination for the texture recognition problem. LTP uses a threshold to generate three values for the pixel difference in the given image. The threshold value is a choice of the user in LTP which makes it resistant to noise. LTP was used successfully along with other descriptors to solve face recognition

problem under varying lightning conditions.

Novel Extended Local Binary Pattern (NELBP): NELBP descriptor was introduced by Zhou et al. in 2008 [54]. The main advantage of NELBP over the original LBP is that it makes use of the nonuniform patterns of texture features based on their occurrence probability. Based on a similarity measure, NELBP assigns each nonuniform pattern to the corresponding uniform pattern. Rotation invariance of the new descriptor was also considered and the descriptor proved to be efficient and robust against noisy pixels in the image. However, NELBP has a shortcoming of not providing a good classification accuracy when applied to a smaller neighborhood compared to LBP. Authors of NELBP justifies this disadvantage by stating that nonuniform texture patterns have small frequency of occurrences in the image.

Pairwise Rotation Invariant Cooccurrence Local Binary Pattern (PRICoLBP): PRICoLBP descriptor was introduced by Qi et al. in 2014 [55]. The new descriptor is based on the cooccurrence features concept but with better robustness and less sensitivity to geometric variations. The design of the new features also involves the rotation invairance, multi-scale, and multi-channel information. PRICoLBP was evaluated extensively on nine different benchmarks and applied to six different applications including: texture, flower, material, food, leaf, and scene classification. The new features proved to be effective and powerful and provide a good discrimination and robustness tradeoff. One important property of PRICoLBP descriptor which allows this superior classification performance is the size of the feature vector. PRICoLBP has a feature vector of size 3540 bins. When color is added to further improve the performance, the total dimension of the descriptor is 10620. In the design of our features, we created two descriptors and considered all the possibilities to extract and make use of all the texture features in the image in order to improve the classification accuracy.

Joint-scale LBP (JLBP): JLBP descriptor was introducd by Wu et al. in

2017 [56]. A new method of encoding texture patterns is introduced where not only micro-texture regions are described, but also, the macro-textures of a larger region making use of the joint multiple scales property for extracting the texture features. To further enhance the power of JLBP, a Completed modeling CJLBP of the descriptor was also used the same way as in CLBP [48] descriptor. Evaluation of JLBP descriptor on benchmark texture databases using the NN classifier proved that the new method of joining multi-scale LBP patterns are very efficient and can improve the classification accuracy.

Binary Rotation Invariant and Noise Tolerant Texture descriptor (BRINT): In 2014, Liu et al. proposed a very robust, combact, and fast to built descriptor called BRINT [57]. BRINT uses a combination of three different operators: BRINT S, BRINT M and BRINT C. In addition, BRINT not only uses rotation invariant "uniform" patterns as in CLBP, but, it uses all of the rotation invariant patterns. The pixels in BRINT are sampled in a circular neighborhood in a way that keeps the number of bins in a single-scale LBP histogram constant and small. BRINT does not require any dictionary building or tuning parameters such as the methods that rely on clustering. Extensive experiments on standard databases show that the performance of BRINT is both superior and robust in the presence of noise.

Dominant Rotated Local Binary Patterns (DRLBP): DRLBP was introduced by Mehta et al. in 2016 [58]. The descriptor overcomes the problem of fixed weights arrangements in the original LBP descriptor. The rotation invariance in DRLBP is achieved by calculating the operator with respect to a fast computed reference in a local neighborhood. DRLBP not just preserves the complete structural information, but also makes use of the magnitude information neglected by LBP. However, the calculation of the descriptor involves learning a dictionary of the most frequently occurring patterns from the given training images. Experimental results show that the descriptor performs better than most of the state-of-the-art LBP vari-

ations using NN classifier on standard texture databases.

Rotation Invariant co-occurrence among Local Binary Patterns (RIC-LBP): In 2014, Nosaka et al. proposed a new variation of LBP descriptor called RIC-LBP [2]. RIC-LBP was introduced to classify HEp-2 cell images. It makes use of the relationships among the binary patterns by finding the co-occurrences patterns among the histogram features. RIC-LBP will be discussed in details when we introduce our framework since we benefit from these features in both texture classification databases and HEp-2 cell and specimen classification.

Joint adaptive median binary patterns (JAMBP): JAMBP descriptor was proposed for texture classification in 2015 by Hafiane et al. [3]. The descriptor is based on the adaptive median filter, where the center pixel in a specific image neighborhood is replaced with the median value of that region. Hence, a new descriptor is created called Adaptive Medina Binary Pattern (AMBP) which is robust against noise by definition. Moreover, in order to enhance the feature extraction power of AMBP, additional information are jointly combined with AMBP features. These information represent the mean of the image and the window size used around each pixel to compute the median value. More details on how this descriptor operates will be discussed in details when we introduce our framework for texture classification.

Extended Local Binary Pattern (ELBP) ELBP descriptor was proposed in 2012 by Liu et al. [59]. ELBP descriptor is inspired by LBP descriptor where four different descriptors are used to perform texture classification. The features of these descriptors are both pixel intensities and differences extracted from local neighborhoods. Two of these four descriptors represent the intensity-based features of the central pixel and the neighbor pixels. While the other two descriptors represent difference-based features which are the radial-difference and the angular-difference. ELBP descriptor is easy to implement and requires no learning step or the use of texton dictionary. The descriptor proved to be efficient for texture classification task

using standard texture databases.

Dominant Local Binary Pattern (DLBP): DLBP descriptor was introduced in 2009 by liao et al. [60]. The features of DLBP descriptor involve two sets: the dominant LBP features in the texture images and additional features which represent the responses of the circularly symmetric Gabor filter. The first set of features can be understood as the most frequent features in the image to describe textural information. While the Gabor filter responses features represent global textural information extracted from the image. Thorough experiments on standard texture databases demonstrate that DLBP descriptor achieves high accuracy compared to state-of-the-art descriptors.

2.2 Texture Features Extraction using Motif Cooccurrence Matrix

2.2.1 Space Filling Curve Concept

In mathematics, space filling curve concept can be defined as a path of a continuously moving point in 2 (or higher) dimensions as illustrated in Figure 2.3. Hence, the range of this curve contains the entire 2-dimension square unit (or more in higher dimensions). Since Giuseppe Peano was the first to discover such continuous curve, space filling curves now are referred to as Peano curves or Peano scans [61].

The purpose of Peano when he first introduced this space filling concept was to create a continuous mapping from the unit interval (in mathematics, unit interval is the closed interval [0,1]) onto the unit square (in mathematics, unit square is a square with sides have the value of 1). Peano wanted to prove such a mapping does really exist and it is continuous. Peano's curve was further extended to higher dimensions and to continuous curves without endpoints. The Peano scans are computed recursively

using spatially motifs defined over 2 x 2 grid. Seetharaman et al. [62] constructed a Z-tree to represent 2-D images which is also computed recursively. They showed that Peano scanning is a good alternative to rasterization in transforming the image into a 1-D vector since it is less computationally expensive and leads to a better performance.

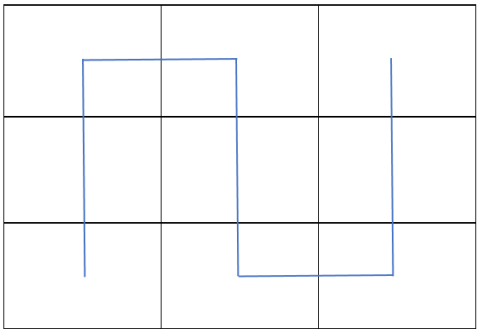


Figure 2.3: Three iterations of the Peano scan construction.

In order to find the optimal or the best effective Peano scan that can traverse the image, Jhanwar et al. searched for that scan and used a specific modulation system called Differential Pulse Code Modulation (DPCM) [1]. The DPCM system is used to transmit a rasterized image (In computer graphics, a rater image is a specific data structure called dot matrix and it is used to represent a grid of pixels or point of color). Hence, Jhanwar et al. found that the compression system would be very effective if the image is made of large number of constant valued streaks or runs. However, there are some limitations in case the image is rotated by a degree in a direction normal to the image plane leading the overall compression to be dropped sharply. The reason is because the rasterizer is unable to scan the data in all directions.

Scanning image pixels locally should be performed such that it minimizes the variation of intensities along the whole line of scan. As a results Jhanwar et al. used one of the six primitive scans as shown in Figure 2.4 to scan the entire image based on a 2 x 2 neighborhood patches. The impact of using a 2 x 2 optimal scan is to

reduce the abrupt variation in intensities along the scan. In this case, the image is to be transformed into a newer form in which the spectral content is concentrated on a narrawer zone.

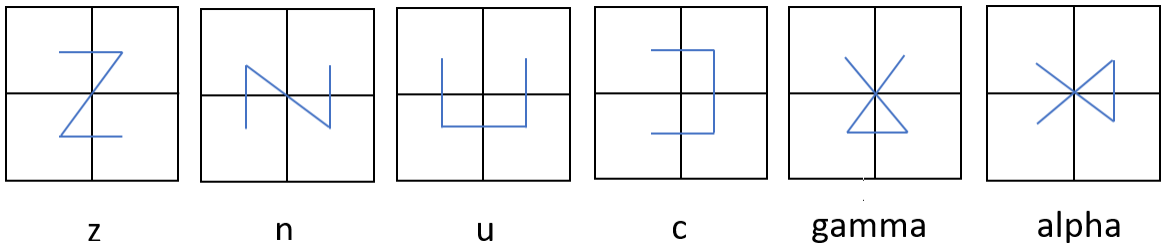


Figure 2.4: The six distinctive motif scans used to traverse a 2×2 image neighborhood.

2.2.2 Motif Cooccurrence Matrix (MCM) [1]

Jhanwar et al. benefited from the space filling curve concept and derived a new texture features that traverses the input image in a specific order through a cooccurrence measure using a set of six Peano scans at a specific distance. The choice of a particular motif over any 2×2 grid depends on the local texture occupying the grid. Then, a statistical feature such as the cooccurrence matrix provides a rich description of the low level semantics. Hence, the motif cooccurrence matrix (MCM) can be assumed to have captured the local statistics in a compact way so that the images can be compared based on their respective MCMs.

Moreover, translational invariance was also achieved to make sure the extracted features are invariant to any rotation or pixel shifting. Instead of calculating a single MCM feature vector, four feature vectors are extracted for each image. They correspond to the MCMs corresponding to the original image, and the images shifted by one pixel horizontally, vertically and diagonally. In this case the spatial relationship of the motif images is preserved in one of the four feature vectors.

2.2.3 Variations of MCM

Modified Color Motif Co-occurrence Matrix MCMCM: MCMCM descriptor was proposed by Subrahamanyam et al. in 2013 for image indexing and retrieval [63]. The proposed method collects the inter-correlation between the red, green, and blue color planes which is absent in color motif co-occurrence matrix. Hence, nine color motifs from the available three color motifs (RGB) are found by considering inter-correlation between RGB color planes. In addition, the proposed method integrates the MCMCM and difference between the pixels of a scan pattern (DBPSP) features with equal weights in contrast to the system which integrates motif co-occurrence matrix, DBPSP, and color histogram with k-mean features with optimized weights. The performance of the proposed method was tested and showed high accuracy compared to other methods.

Adaptive Motifs Co-occurrence Matrix (AMCOM): AMCOM descriptor was proposed in 2011 by Lin et al [64] for image retrieval application. Moreover, another descriptor was also proposed which is based on the motif Peano scan concept called: Gradient Histogram for Adaptive Motifs (GHAM). The AMCOM calculates the distribution within the 2D motifs of scan pattern matrix for Adaptive Motifs of Pattern Block (AMPB). Then, the probability of the co-occurrence of the motif pattern block is used as an image feature. After GHAM is computed, the histogram of the mean gradient of the motifs of pattern block is adopted as an image feature. GHAM estimates the mean gradient of a pattern block to describe the texture of the pattern block.

Lin et al. proposed a framework of four image features for efficient content-based image retrieval [65]. The first and second image features are based on color and texture features, respectively called Color Co-occurrence Matrix (CCM) and Difference Between Pixels of Scan Pattern (DBPSP) in their work. The third image feature is based on color distribution, called color histogram for K-mean (CHKM). A

CCM can be obtained by presenting each image as four matrices of motif patterns, then, the attribute of the original image will be calculated with these motif patterns. CCM could be thought as a representation of the direction of features but not the complexity. On the other hand, DBPSP can determine the complexity by calculating the difference among all pixels within the six motif patterns derived from the image.

Directional Local Motif XoR Patterns (DLMXoRPs): DLMXoRPs descriptor was proposed by Vipparthi et al. in 2014 for image retrieval [66]. The DLMXoRP presents a novel technique for the calculation of motif using 1×3 grids. The proposed motif 1×3 representation is having a flexible structure; hence it is able to extract all directional information. This flexibility is not present in the existing 2×2 motif. Furthermore, the XoR operation is performed on the transformed new motif images which are not present in the literature (local binary patterns (LBP) and motif co-occurrence matrix (MCM)).

Color Based Multi-directional Local Motif XoR Patterns (CMDLMXoRP): CMDLMXoRP was introduced by Rao et al. in 2015 for image retrieval [67]. Here, the joint correlation between directional smart grid is proposed. First, the required directional information is calculated. In the next stage, smart grid XOR patterns are applied to generate transformed smartgrid images in four directions. This entire operation is implemented on 'V' color space of HSV color plane.

Vipparthi et al. presented a novel method for image retrieval based on the motif Peano scan concept called: Multi-Joint Histogram based Modelling (MJHM) [68]. Here, the joint correlation histograms are constructed between the motif and texton maps. Firstly, the quantized image is divided into non-overlapping 2×2 grids. Then each grid is replaced by a scan motif and texton values to construct the transformed motif and texton maps (images) respectively. The motif transformed map minimizes the local gradient and texton transformed map identifies the equality of gray-scales while traversing the 2×2 grid. Finally, the correlation histograms are constructed

between the transformed motif and texton maps.

Dual Ddirectional Multi-Motif XoR Pattern (DDMMXoRP): DDMMXoRP was proposed by Vipparthi et al. in 2015 for image indexing and retrieval [69]. First, a new 2×2 standard grid is introduced at distance two and four 1×3 smart grids along dual directions which are not present in existing motif representation. This entire operation is implemented on 'V' color space of HSV color plane. Furthermore, the XOR operation is performed on the transformed new motif images.

2.3 Statistical Methods

The previous two sections focused on both LBP and MCM descriptors since our framework for image classification is based on these descriptors. However, there are also other powerful descriptors which proved to be robust to extract and classify texture features.

Scale Invariant Feature Transform (SIFT): SIFT was proposed by Lowe et al. in 1999 and was used successfully for object recognition [70]. SIFT descriptor transforms an image into a large collection of local feature vectors, each of which is invariant to image translation, scaling, and rotation. Moreover, illumination changes are also handled efficiently. SIFT uses a staged filtering approach where key points are identified at the first stage by looking for locations that are maxima or minima of a difference of Gaussian function. Each point is used to generate a feature vector that describes the local image region sampled relative to its scale-space coordinate frame. SIFT proved to be very powerful to classify images based on texture features where eight orientation planes are computed first from the image. Then, the gradient image is sampled over a 4×4 grid of locations resulting in a $4 \times 4 \times 8 = 128$ dimensional feature vector for each region.

Bag of Visual Words (BoVW): In the bag of visual words image classification

model, the feature vectors generated by the keypoint descriptor are grouped into a set of given number of clusters using a vector quantization algorithm such as K-Means [71]. This process forms a codebook which represents the visual features extracted from the training set. The next step consists of representing each image into a histogram of codewords, by first applying the keypoint detector and descriptor to every training image, and then matching every keypoint with those in the codebook. The result is a histogram where the bins correspond to the quantized keypoints in the codebook, also known as codewords, and the count of every bin corresponds to the number of times the corresponding codeword matches a keypoint in the given image. In this way, an image can be represented by a histogram of codewords. The histograms of the training images can then be used to learn a classification model.

Fisher Vector (FV): FV representation of an image is considered as an alternative to the popular Bag-of-Visual words (BoV) encoding technique commonly adopted for the image classification task [17]. Within the Fisher Vector framework, images are characterized by first extracting a set of low-level patch descriptors and then computing their deviations from a "universal" generative model, i.e. a probabilistic visual vocabulary learned offline from a large set of samples. This characterization is given as a gradient vector w.r.t. the parameters of the model, which we choose to be a Gaussian Mixture with diagonal covariances. Compared to the BoV, the Fisher Vector offers a more complete representation of the sample set, as it encodes not only the (probabilistic) count of occurrences but also higher order statistics related to its distribution w.r.t. the words in the vocabulary. The better use of the information provided by the model translates also into a more efficient representation, since much smaller vocabularies are required in order to achieve a given performance.

Leung and Malik Filters (LM): LM filters are a statistical approach that is based on texton concept and were introduced by Varma and Zisserman in 2005 [18]. The LM set consists of 48 filters, partitioned as follows: first and second derivatives of

Gaussians at 6 orientations and 3 scales making a total of 36; 8 Laplacian of Gaussian filters; and 4 Gaussians. The scale of the filters range between $\sigma = 1$ and $\sigma = 10$ pixels. The classification stage of texton methods involves two stages: learning and classification stages. In the learning stage, training images are convolved with a filter bank to generate filter responses. Exemplar filter responses are chosen as textons via K-Means clustering and are used to label each filter response, and thereby every pixel, in the training images. The histogram of texton frequencies is then used to form models corresponding to the training images. In the classification stage, the same procedure is followed to build the histogram corresponding to the novel image. The histogram is then compared with the models learned during training and is classified on the basis of the comparison.

MR8 Filters [21]: The MR8 filter bank consists of 38 filters, however, only 8 filter responses are considered. These bank of filters include filters at multiple orientations and their outputs are collapsed by recording only the maximum filter response among all orientations. As a result, these filters are able to achieve rotation invariance. The components of these filters include both Gaussian and Laplacian of Gaussian filters with different orientations and scales. MR8 filters can help to reduce the effect of rotation invariance filters which do not respond efficiently to oriented image patches that leads to providing poor features. In order to do this, MR8 provide both isotropic and anisotropic filters. Moreover, MR8 filters are capable of recording the angle of the maximum response which leads to compute higher order co-occurrence statistics orientation and provide better texture discrimination.

Gray-Level Co-occurrence Matrix (GLCM): GLCM was proposed by Haralick et al in 1973 [5] and is considered as one of the earliest methods for texture feature extraction. The GLCM functions characterize the texture of an image by calculating how often pairs of pixel with specific values and in a specified spatial relationship occur in an image. GLCM descriptor considers different directions to analyze the image

including: horizontal, vertical, and diagonal direction. For example, in order to build a traditional Co-occurrence matrix, we can assume I to be a given grey scale image. Let M be the total number of grey levels in the image. The Grey Level Co-occurrence Matrix defined by Haralick is a square matrix G of order M, where the $(k, m)^{th}$ entry of G represents the number of occasions a pixel with intensity k is adjacent to a pixel with intensity m. The normalized co-occurrence matrix is obtained by dividing each element of G by the number of co-occurrence pairs in G.

2.4 Deep Learning Methods for Texture Representation

Deep learning is a division of machine learning methods based on learning data representations in which learning can be supervised or unsupervised [72, 73]. Many applications benefited from deep learning including speech recognition, biomedical image analysis, natural language processing, social network filtering, audio recognition, machine translation, bioinformatics. The popularity of deep learning in recent years came from its ability to generate accurate results and sometimes outperform human experts in certain fields. In general, neural networks are composed of several layers which are made of nodes. Inside nodes, many computations happen where it accepts input from the data along with some parameters like weights which can be thought as assigning significance to these inputs for the task under learning. After that, these input-weight data are summed and the sum is passed through an activation function, to determine whether and to what extent that signal progresses further through the network to affect the ultimate outcome, for example, an act of classification.

The main property that distinguishes deep learning networks from other neural networks with signle layers is the depth. In other words, the number of node layers

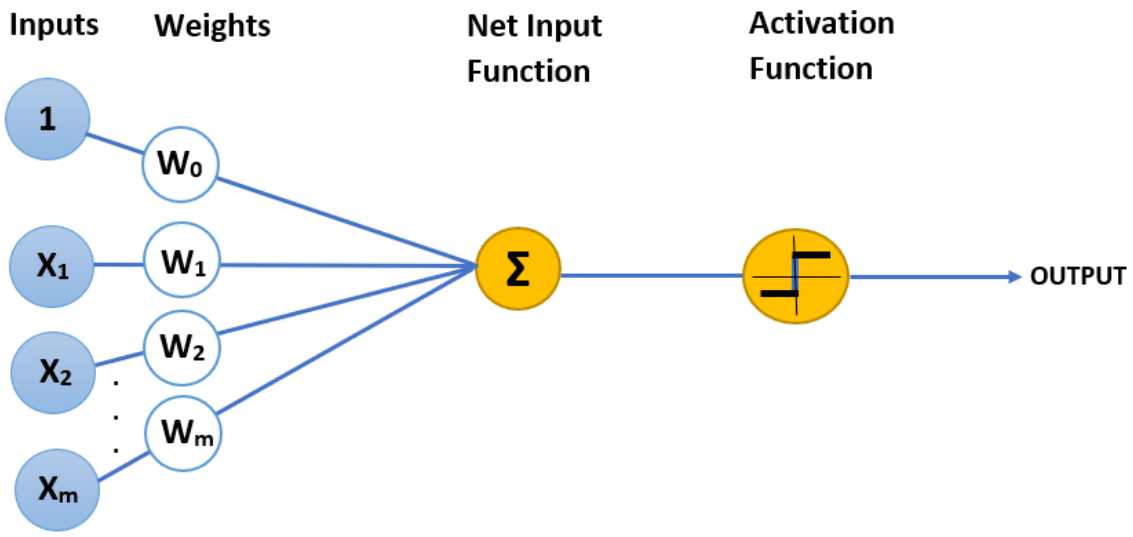


Figure 2.5: A node layer is a row of switches that turn on or off as the input is fed through the net. Starting from an initial input layer that receives your data, each layer's output is simultaneously the subsequent layer's input.

through which multi-step process of pattern recognition operations will be conducted on data. At the beginning, neural networks such as the first perceptrons were composed of one input and one output layer, and at most one hidden layer in between [74]. If the network was composed of more than three layers (including input and output), it is qualified as "deep" learning.

During training of each layer of nodes in deep learning, a distinct set of features is used based on the output of the previous layer as shown in Figure 2.5. In addition, neural network recombine and aggregate features from previous layers, as a result, more complex features can be recognized as we go further in the network. This property of deep learning makes it capable of handling high dimensional and very large datasets with billions of parameters that pass through the network layers. A simple example can be seen in clustering a million images according to their similarities using the deep learning technique [75]. Moreover, deep learning can be used in other unsupervised learning tasks like clustering emails or news articles. Voice messages can also be clustered in a similar manner. Hence, data might cluster around anomalous/dangerous behavior and normal/healthy behavior which will provide insight into

users' health and habits [76].

The steps involved in training a neural network are [77]:

- **Initialize weights and biases**: The weights and biases can be initialized to zeros, in this case, the model will be linear. Hence, the derivative with respect to loss function will be the same for every weight. In addition, hidden unites will be symmetric for all iterations during training. In general, setting biases to zeros will not cause any issue. Another way to initialize weights is by considering random values following a standard normal distribution. In this case, weights can be initialized close to zero which will help in breaking symmetry at every neuron.

- **Forward propagation**: In deep neural network, the input data progresses in the forward direction through the network. Using the input X, weights W and biases b, for every layer we compute y. At the final layer, we compute a special function $f(y^{(L-1)})$ which could be a sigmoid, softmax or linear function of $A^{(L-1)}$ and this gives the prediction $\hat{y}$.

- **Compute the loss function (error function or cost function)**: This is a function of the actual label y and predicted label $\hat{y}$. It is used to indicate how far off our predictions are from the actual target. Finally, our objective is to minimize this loss function in order to get better prediction accuracy.

$$LossFunction = 1/2(\hat{y} - y)^2 \tag{2.2}$$

- **Backward Propagation**: In this step, the gradients of the loss function $f(y, \hat{y})$ will be calculated with respect to A, W, and b and will be called dA, dW and db. These gradients will be used to update the values of the parameters from the last layer to the first layer. Steps 2 –4 are repeated for n iterations/epochs

until we feel the loss function is minimized enough without overfitting the train data.

2.4.1 Components of Convolutional Neural Networks

In order to understand the components of modern CNNs, we selected a well known architecture called AlexNet [22] which was introduced in 2012 and achieved high performance on ILSVRC-2012 competition compared to other techniques. Authors managed to train AlexNet to classify 1.2 million high-resolution images provided by the competition into 1000 different classes. In addition, their network consists of 60 million parameters and 650,000 neurons with five convolutional layers and five fully-connected layer with a final softmax layer. This work is considered the benchmark for deep learning and many of the architectures that followed AlexNet used the same concepts but with deeper networks.

We can summarize the important components of current CNNs as follows:

- **Pooling**:

  Pooling is considered the backbone of the current success of deep learning architectures. It works by converting a vector of data into a scalar which operates on each region of the image. In pooling, there are no filters and no dot products computations with respect to local regions, instead, pooling compute the average of the pixels in the region (Average Pooling). Another way of pooling is by picking the pixel with the highest intensity and discards the rest (Max Pooling). That means, the size of the feature map will be reduced. Although the idea seems to lead to information loss, it proved to be very effective in practice since it reduces the effect of background noise and makes the network invariant to variations in the presence of an image.

  On the other hand, Max Pooling proved to work very well in recent years. As

mentioned above, it is based on the idea that the maximum pixel in a region represents the most important feature in that region. In general, when we want to classify image of a specific object, there could be other objects in that image. For example, a classifier could classify a car image to be a cat image if there exist a cat along with the car at that same image. As a result, the overall classification accuracy could be degraded. Pooling helps to alleviate this kind of effects, and makes ConvNets generalize better. For AlexNet architecture, overlapping pooling is used to reduce the effect of overfitting and to reduce the overall error.

- **Dropouts**:

  Overfitting can easily causes a CNN to work well in training set of images while performing poorly on the testing data. To reduce the effect of overfitting Dropout technique can be used during CNN training phase. Dropout works by randomly setting some activations to 0, basically eliminating them. By doing this, the network is forced to explore more ways of classifying the images instead of depending largely on some specific features. For example, AlexNet used the dropout technique which was introduced in 2012 by Hinton et al. [78]. Hence, The neurons, which are dropped out, do not contribute to the forward pass and in backpropagation. As a resulot, every time an input is presented, a different architecture is sampled by the network. However, all these architectures share weights. In this case, the network is forced to learn more robust features that could be more useful.

- **Batch Normalization**:

  Another major problem with CNN is called vanishing gradients. When it happens, gradients will be too small which will lead to a big issue in the training process. Researchers from Google [79] discovered that internal covariate shift

in data is to blame for this issue. This problem can be solved by introducing a technique known as batch normalization which works by having every image batch to have zero mean and unit variance. Batch normalization usually occurs before non-linearity (ReLU) in CNNs. The accuracy can be greatly improved and it also helps in speeding up the training process.

- **Data Augmentation**

  In modern CNNs, one of the most important ingredient is data augmentation. This is because it is difficult for machines to adapt to image rotations, translations, and other distortions. To solve this problem, images are randomly distorted and rotated before the training stage. In this case, CNNs will learn how to handle these distortions, hence, they would be able to work well in the real world. In AlexNet, authors employed two different forms of data augmentation. The first form of data augmentation is to generate image translations and horizontal reflections. This was done by extracting random 224×224 patches from the given images and the network was trained using these extracted patches. This augmentation caused the size of the training set to be increased by a factor of 2048 in the case of AlexNet.

  The second form of data augmentation consists of altering the intensities of the RGB channels in training images. Specifically, PCA was performed on the set of RGB pixel values throughout the ImageNet training set. In this case, for each image in the training set, principle components were added with magnitudes proportional to the corresponding eigenvalues times a random variable drawn from a Gaussian with mean zero and standard deviation 0.1.

2.4.2 CNNs for Texture Representation

After the success of AlexNet [22], large number of CNN-based texture representation methods have been developed in the recent years. As we detailed the components of CNN section, the main advantage of these networks is their ability to handle large labeled datasets and learn high quality features. In addition, it was found that CNN features pretrained on very large datasets can transfer well to many other problems, including texture analysis [23, 24, 26, 27]. In general, CNN for texture representation can be divided into the following categories:

- Pretrained CNN models.
- Finetuned CNN models.
- Handcrafted deep convolutional networks.

Pretrained CNN Models

Feature extraction and encoding steps of generic pretrained CNN laid the way to the success of these models in texture representation. Successful networks for pretraining and feature extraction include the following models:

- **Popular CNN Models**: These models can be exploited in extracting features like AlexNet [22], VGGNnet [4], GoogleNet [80], ResNet [81], and DenseNet [82]. Evaluations of feature transfer effect of CNNs have been studied for the texture classification task [83, 24, 84], and the following insights were found. features extracted from convolutional layers or fully connected layers have shown varying classification performance during model transfer. Experiments proved that the fully-connected layers of the CNN tend to generate worse generalization ability and transferability, and therefore would need retraining or finetuning on the transfer target. The convolutional layers on the other hand usually transfer

well. As a result, the source training set is relevant to classification accuracy on different datasets. Moreover, it was also found that deeper models like VGGNet and ResNet transfer better.

- **Feature Extraction**: It was found that the useful approach to CNN-based texture classification is to extract features from the fully connected layer e.g., 'fc6' or 'fc7 [83, 24]. The features of fully connected layers have a global receptive and can be considered as global features suitable for classification with any machine learning classifier like SVM, RF, or kNN. On the other hand, the features of the convolutional layers of CNN can be used as filter banks to extract local features. Compared with the global fully-connected features, lower level convolutional features are more robust to image transformations such as translation and occlusion.

- **Feature Encoding and Pooling**: Features extracted from either convolutional or fully connected layers can be encoded using any technique like FV [85], VLAD [86], LLC [87], BoW [20] as done by Cimpoi et al. [24]. After that, Song et al. [88] proposed a network to transform FVCNN descriptors to lower dimensional representation. Recently, Gatys et al. [89] showed that the Gram matrix representations extracted from various layers of VGGNet can be inverted for texture synthesis. In addition, the bilinear feature pooling, which is an orderless pooling representation of the input image, is suitable for texture modeling. The Bilinear CNN (BCNN) features are obtained by computing the outer product of each feature with itself and reordered into feature vectors. After that, pooling can be used by summing to obtain the final global representation [90].

Finetuned CNN Models

Pretrained CNN models have achieved superior performance in recognizing texture. The only disadvantage is that the the training stage using these models requires many steps: feature extraction, codebook generation, feature encoding, and classifier training. Generally finetuning CNN models on task-specific training datasets (or learning from scratch if large-scale datasets are available) is expected to improve on already strong performance achieved by pretrained CNN models [25]. When finetuning a CNN, the last fully connected layer is modified to have N nodes corresponding to the number of classes in the target dataset. It is worth to mention that Andrearczyk and Whelan [91] observed that finetuning a network that was pretrained on a texture-centric dataset achieves better results on other texture datasets compared to a network pretrained on an object-centric dataset of the same size. Gao et al. [92] proposed compact bilinear pooling, which utilizes Random Maclaurin Projection or Tensor Sketch Projection to reduce the dimensionality of bilinear representations while maintaining similar performance to the full BCNN feature [93] with a 90% reduction in the number of learned parameters.

Handcrafted Deep convolutional Networks

There are some handcrafted deep learning networks that deserve attention including the Scattering convlultion Network (ScaNet) proposed by Bruna and Mallat [94]. The key difference between these networks and CNN is that convolutional filters in ScaNet are predetermined (since they are wavelet filters such as Gabor or Haar wavelets) and no learning is required. Moreover, the ScatNet usually cannot go as deep as a CNN. Hence, Bruna and Mallat suggested two convolutional layers, since the energy of the third layer scattering coefficients is negligible. The average pooled feature vector from each stage in ScaNet is concatenated to form the global feature representation of an image, which is input into a simple PCA classifier for recognition. Somewhat

surprisingly, such a prefixed network has demonstrated very high performance in texture recognition [94, 95, 96]. A downside of ScatNet is that the feature extraction stage is very time consuming, although the dimensionality of the global representation feature is relatively low.

2.5 HEp-2 Cell and Specimen Classification Techniques

Pattern recognition techniques are widely used in the field of medicine for the development of Computer-Aided Diagnosis (CAD) systems. Such systems may support the physician in many ways: they can be adopted as a second reader, thus augmenting the physician's capabilities and reducing errors; they make it possible to perform a preselection of the cases to be examined, enabling the physician to focus the attention only on the most relevant cases and hence facilitating mass screening campaigns. They also may aid the physician in carrying out the diagnosis; finally, they can be used as a tool for the instruction and training of specialized medical personnel [97].

Among such applications, over the last few years there has been a certain interest in the realization of CAD systems for the analysis of indirect immunofluorescence (IIF) images. IIF is a diagnostic methodology based on image analysis that reveals the presence of autoimmune diseases by searching for antibodies in the patient serum. As a result of its effectiveness, we have witnessed a growing demand of diagnostic tests for systemic autoimmune diseases. Unfortunately, however, IIF as yet remains a subjective method that depends too heavily on the experience and expertise of the physician.

In order to classify the fluorescence intensity, the guidelines established by the Center for Disease Control and Prevention(CDC), Atlanta, GA, USA [98] suggest scoring it semi-quantitatively and independently by two physicians who are experts

in IIF. The score ranges from 0 up to 4 with 0 considered negative and 4 considered brilliant green. Since technical problems can affect test sensitivity and specificity, the same guidelines suggest comparing the sample with a positive and a negative control. The former allows the physician to check the correctness of the preparation process, whereas the latter represents the auto-fluorescence level of the slide under examination.

The variability between a set of physician's fluorescence intensity classifications were statistically analyzed by Rigon et al. [99]. After that, Rigon proposed to classify fluorescence intensity into three classes: negative, intermediate, and positive. Theses classes maintain the clinical significance of IIF testing and establish a more robust ground truth. Finally, the staining pattern recognition is important to be achieved. This is a very challenging task as many patterns, corresponding to different autoimmune diseases, may be observed. The most frequent patterns are:

- **Centromere**: defined by many discrete speckles (40−60) distributed throughout the interphase nuclei and characteristically found in the condensed nuclear chromatin during mitosis as a bar of closely associated speckles.
- **Nucleolar**: defined by large granules in the nucleoli of interphase cells which tend towards homogeneity, with less than six granules per cell.
- **Homogeneous**: characterized by a diffuse staining of the interphase nuclei and staining of the chromatin of mitotic cells.
- **Fine Speckled**: characterized by a fine granular nuclear staining of the interphase cell nuclei.
- **Coarse Speckled**: characterized by a coarse granular nuclear staining of the interphase cell nuclei.

In recent years, great strides have been made towards obtaining automatic pattern recognition and image analysis of HEp-2 cell and specimen images due to the international contest conducted to tackle this problem. Many methods were introduced in these contests with a huge focus on feature extraction and classification. In the following, we will discuss in depth the most successful methods that were introduced in these contests.

2.5.1 HEp-2 ICPR 2012 Competition

The first HEp-2 cell classification contest was held in 2012 in conjuction with the International Conference on Pattern Recognition (ICPR). The contest received 22 papers with significant focus on feature extraction techniques. For this contest, HEp-2 images were acquired by means of a fluorescence microscope (40-fold magnification) coupled with a 50W mercury vapor lamp and with a digital camera. The camera has a chargecoupled device with square pixels of 6.45 . The images have a resolution of 1388×1038 pixels, a color depth of 24 bits and they are stored in an uncompressed format.

As we mentioned earlier, most of the methods submitted for these contests relied on feature extraction techniques. These techniques include: LBP, GLCM, Robust Structure Tensors-Histogram of Oriented Gradients (ARST-HOG), shape index, and filter banks [100, 101, 2, 102]. In addition, most of these methods used k-Nearest Neighbor (k-NN) classifier and Support Vector Machine (SVM) in the classification stage [103]. Others, like Malone et al. used neural network classifier. The top 3 recognition accuracy methods for this contest are as follows [97]:

- Co-occurrence Among Local Binary Pattern (CoALBP) proposed by Nosaka et al. which uses the green channel [104]. Here, the image is filtered by a Gaussian function for noise reduction. Complex texture regions are extracted

using CoALBP descriptor which is a powerful extension of the original LBP. The classifier used is a linear SVM trained with a learning set including the various rotated patterns of the original images.

- Xiangfei et al. proposed a system that adopts Varmas' MR8 method to extract statistical intensity features. Before calculating filter responses, the local regions where the filter is convolved are normalized. A global texton dictionary is trained using K-means clustering. Then each image is represented by the frequency histogram of the textons. The adopted classifier is a k-NN.

- Kuan et al. used four texture descriptors to recognize HEp-2 cells. These descriptors are: a rotation invariant form of LBP with multi-scale analysis, DCT, the mean values and standard variances of 2D Gabor wavelets, and some global appearance based statistical features. A multiclass posterior probability SVM is utilized on each of the four feature sets [102].

2.5.2 HEp-2 ICPR 2014 and 2016 Competitions

Two contests were held after that in 2014 and 2016 as part of the ICPR which included two main tasks: cell classification and specimen classification. Methods submitted for cell classification task used local feature extraction, feature encoding, and deep learning techniques. Local descriptors adopted in this task included: LBP, motif features, and dense scale-invariant features [105, 106]. Methods with feature encoding adopted various local features like: SURF, SIFT, LBP, and cooccurrence of adjacent LBPs and used bag of visual words and Vectors of Locally Aggregated Descriptors (VLAD) to encode them [107, 108]. For deep learning methods, Jia et al. [109] extracted CNN features from a deeper network architecture and used these features to classify ICPR 2016 HEp-2 cell images with an accuracy of 98.26%. For the specimen level task, both 2014 and 2016 competitions used the seven class speciment

cell images of I3A 2014 Task 2 competition. Voting methods and morphological features were used by [107]. Prasath et al. [106] also used RIC-LBP descriptor and achieved an accuracy of 73.43% using RF classifier with 500 trees. Li et al. [110] used a fully convolutional network (FCN) adapted from VGG-16 and achieved a classification accuracy of 90.89%.

It is easy to notice that most of the previous work to classify HEp-2 cell and specimen level were achieved using advanced hand-crafted local features (e.g. multi-resolution local patterns with cell pyramids as in our entries [111], dense scale-invariant descriptors [112], and CoALBP [104]). This was the motivation for us to try and develop new hand-crafted features to address this problem and to be used in general texture recognition problems as a robust framework for image classification.

Chapter 3

Local and Deep Features Framework for Texture Classification

In this chapter, we introduce our set of local features used to extract texture patterns for general texture classification purposes and for biomedical image analysis application. In addition, we introduce our final framework that incorporates robust deep features extracted from the fully connected layer of a powerful deep learning architecture as illustrated in Figure 3.1. In the previous chapter, we demonstrated the local descriptors that have been used for image classification along with variations of the original proposed techniques like LBP and MCM descriptors. We have also emphasized that the most important stage in image classification is the feature extraction stage where many descriptors have been proposed and developed in the past while the classifiers used in the classification step are almost fixed. Figure 3.2 gives an illustration of these steps. The local descriptors used in our framework for image classification relies on a powerful variations of LBP descriptor and on a new features that are based on the motif Peano scan concept which MCM descriptor is based on. Two variations of LBP descriptor were used which are: RIC-LBP and

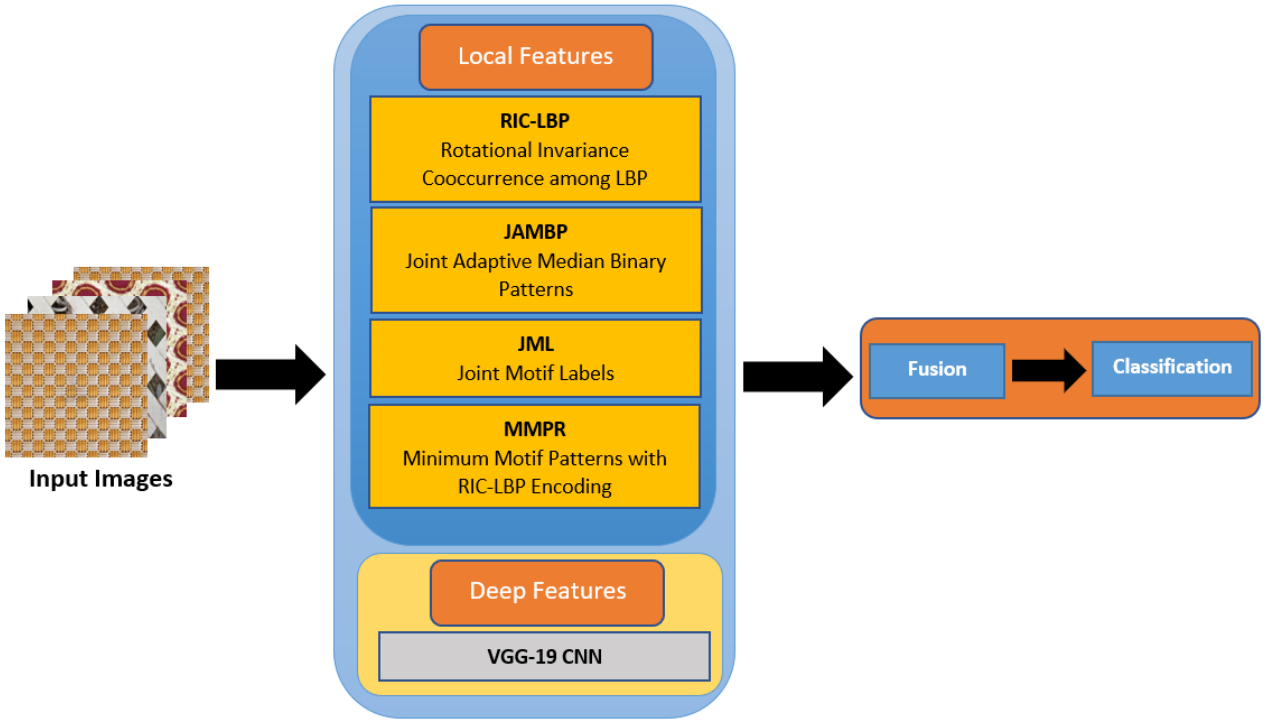


Figure 3.1: Set of local and deep features proposed in our approach. After reading the image, four local features are extracted: RIC-LBP, JAMP, JML, and MMPR. Then, deep features are extracted from 'fc7' layer of VGG-19 architecture. Finally, all these features are fused together by concatenation and a classifier is used (like RF, NN, or SVM) to perform the final classification step and get the accuracy.

JAMBP descriptors. RIC-LBP proved to be efficient for classifying HEp-2 cell images by considering the co-occurrence among the binary patterns. On the other hand, JAMBP also proved to improve the performance of the texture classification task by considering the adaptive median value of the local patch used to compute the AMBP descriptor and by joining these patterns with mean and window scale of the AMBP descriptor. The other two descriptors are derived from the MCM descriptor and are called Joint Motif Labels (JML) and Motif Patterns (MP). Both descriptors are based on the motif Peano scan concept that traverses image pixels in a 2 x 2 neighborhood using 12 distinctive motif patterns. JML uses only the labels of the motif patterns with joint information representing mean and variance of the original image. MP uses the actual motif patterns which will be encoded later with one of the LBP-based descriptors.

For the deep learning features, we extract 4096 features from 'fc7' layer of VGG-19 architecture. After that, we train another classifier with these features and use it to predict the accuracy of our test data. We will show that these combination of features can achieve high accuracy especially to classify biomedical images.

Next, we will detail each step of our frame work and demonstrate the dimensions of our newly developed local features.

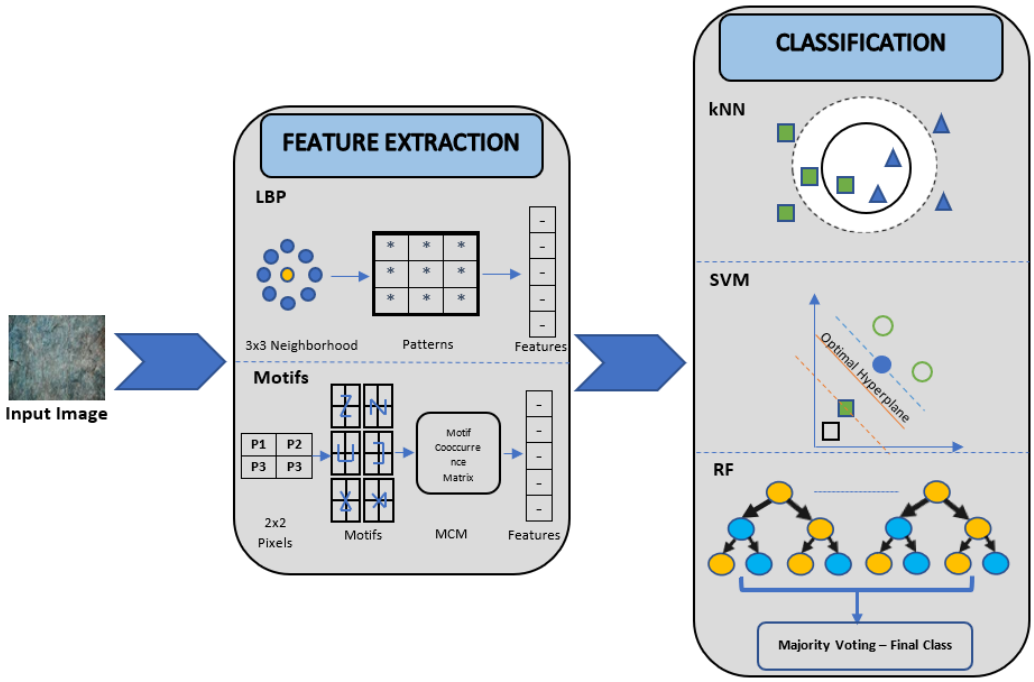


Figure 3.2: Image classification stages: Feature extraction and classification. Recently, the majority of research focus on the first stage, the feature extraction. Many local descriptors have been proposed in the past. In addition, features from different deep learning layers can also be extracted and used in the classification stage.

3.1 RIC-LBP Descriptor [2]

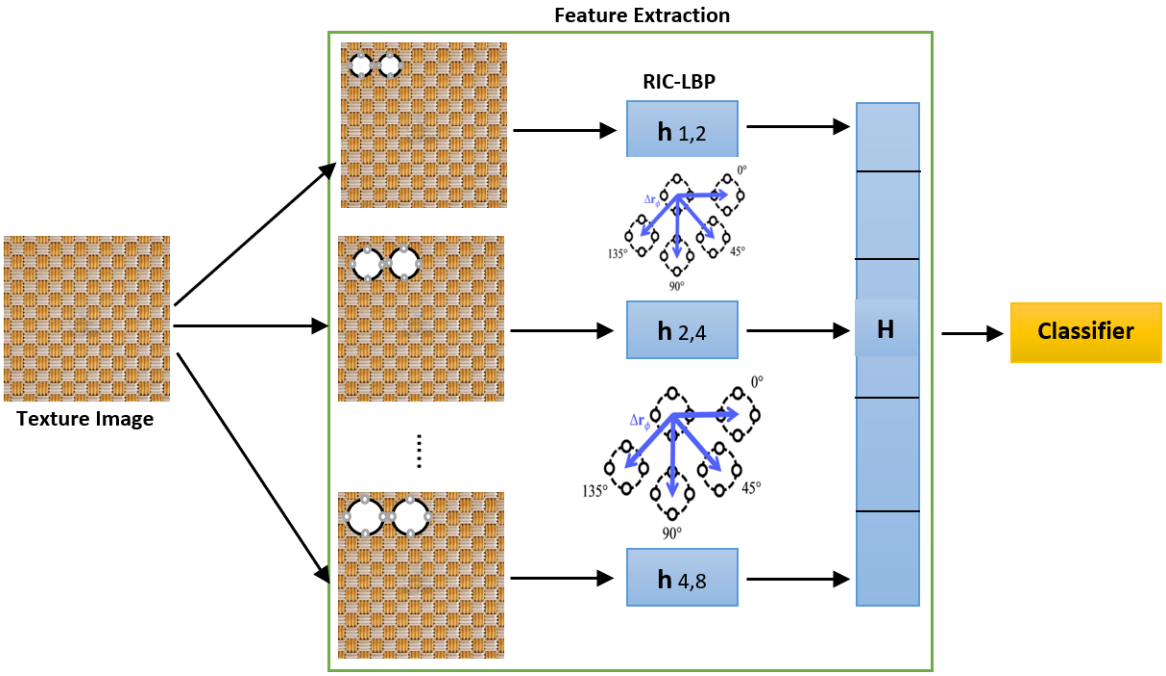


Figure 3.3: Operations performed using RIC-LBP descriptor. The spatial relationships among LBP patterns are found among three different radiuses. Finally, a concatenation of theses features is performed to get the final histogram.

In [2], a new texture descriptor called Rotation Invariant Co-occurrence among LBP (RIC-LBP) was proposed and used successfully to classify HEp-2 cell images. RIC-LBP benefits from the spatial relationships among the binary patterns by finding the co-occurrences patterns among the histogram features which are neglected in the original LBP descriptor. As a result, RIC-LBP histogram will be represented in the form of many LBP pairs and each pair will be attached with a specific label to account for rotation invariance. LBP pairs used in RIC-LBP are demonstrated in Figure 3.4.

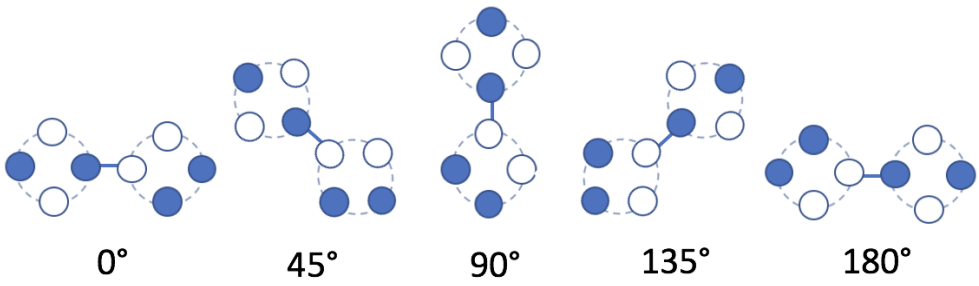


Figure 3.4: Set of equivalent LBP pairs used in RIC-LBP descriptor calculations.

RIC-LBP provides high descriptive ability and robustness against local rotations of an input cell image. To further deal with global rotation, Nosaka et al. synthesize many training images by rotating the original training images and constructing the SVM using both the original and synthesized images. Finally, the feature vector size of RIC-LBP is 408 bins which are extracted in a low computational cost and used in the classification stage as shown in Figure 3.3. Since RIC-LBP is used with gray-scale images only, the original RBG images had to be converted to gray-scale by considering the green channel as an input for the feature extraction stage.

3.2 JAMBP Descriptor [3]

In [3], Hafiane et al. proposed a powerful descriptor that is based on the Adaptive Median Binary Pattern filter called Joint Adaptive Median Binary Pattern (JAMBP). JAMBP is composed of AMBP descriptor jointly combined with other information that represent the mean of the image and the window size used around each pixel to compute the median value. Instead of using a fixed size window to find the median value, AMBP descriptor uses an adaptive window that changes size in order to capture texture features efficiently as shown in Figure 3.5. In this case, AMBP uses the median values of a small patch instead of the center pixel which makes the descrip-

tor more robust in the presence of noise in the original image. Moreover, JAMBP uses multiscale scheme by computing AMBP descriptor using different ranges and sampling points from the center pixel. In addition, it uses a multiresolution scheme by downsampling the image and computing features on the original image and the subsampled one. Finally, the feature vector size used in the classification stage is typically 320 bins. However, JAMBP descriptor is resilient and it can be computed over small image resolutions and increase the feature vector size to account for large scale databases. In general, JAMBP showed a high performance for the texture classification task using standard texture databases.

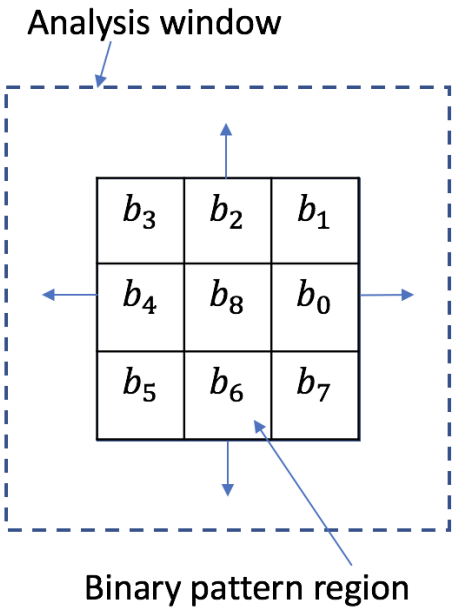


Figure 3.5: Adaptive median binary pattern window. The median value can be found in a larger window like 5×5 instead of a 3×3 window. This has an effect of capturing more texture patterns which leads to a better classification accuracy.

3.3 Joint Motif Labels (JML) Descriptor

3.3.1 Motif Labels (ML)

In the previous chapter, we demonstrated the importance of motif Peano scan concept in providing good representation of image texture features. We also showed that six scan motifs cab be extracted around a 2×2 neighborhood starting from the upper left pixel and a total of 24 possible motif scans can be extracted in this case using the four image pixels. However, only 12 of these scans are distinctive as shown in Figure 3.6. The Optimum peano scan can be found by minimizing the variation of intensity pixels as follows:

$$\delta = |p1 - p2| + |p2 - p3| + |p3 - p4| \tag{3.1}$$

where $p1$, $p2$, $p3$, and $p4$ in Eq. 3.1 correspond to the **Z** motif pixel as shown if Figure 3.6 below.

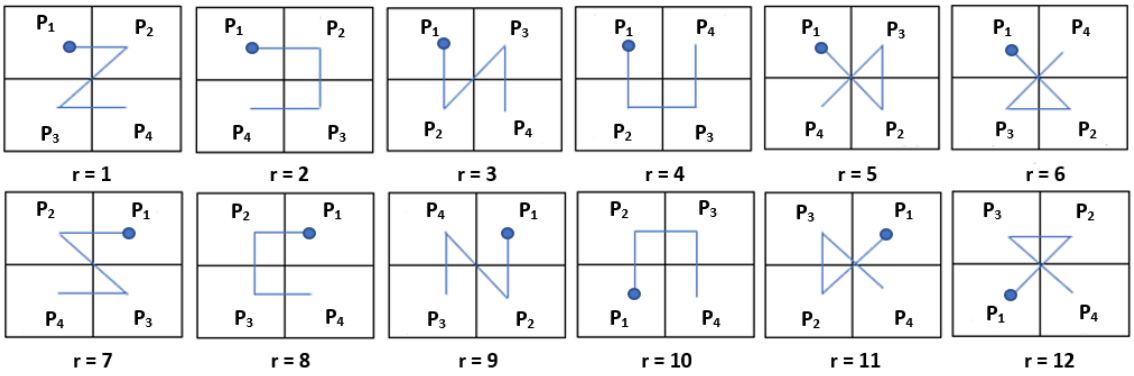


Figure 3.6: The 12 motif patterns used in our approach.

In this section, we derive new features that are also based on the motif Peano scan concept called Joint Motif Labels (JML). The **First** step in computing the JML descriptor starts by traversing each intensity pixel of the input grey scale image and extracting a 2×2 neighborhood patch. In the **Second** step, the 12 distinctive motif

Algorithm 1 Motif Labels.

Input: Grey scale image I.
Output: Motif Labels: Min_ML, Med_ML, and Max_ML.

for all i,j **do**
  $S \leftarrow I[i : i+1, j : j+1]$
  $k \leftarrow 1$
  repeat
    $MPL(k) \leftarrow \delta_k(S)$ - According to Eq. 3.1.
    $k \leftarrow k+1$
  until $k > 12$
  $Min\_ML(i,j) \leftarrow Loc(min(MPL))$
  $Med\_ML(i,j) \leftarrow Loc(med(MPL))$
  $Max\_ML(i,j) \leftarrow Loc(max(MPL))$
end for

patterns are found from this patch using Eq. 3.1 and the corresponding values are stored in 12 separated matrices. Now, each matrix holds the corresponding pattern of the 12 motifs. **Third**, we label each extracted pattern from 1 - 12 as illustrated in Figure 3.7. Since we stored all similar patterns in a separate matrix, this is corresponding to labeling the 12 pattern matrices extracted previously. **Finally**, we find the min, med, and max values of the 12 patterns extracted from each neighborhood and the corresponding label of each pattern is stored in a separate matrix called the Motif Labels matrix. The following equations summarize how to find these motif labels and patterns:

1. Minimum Motif Patterns and Labels:

$$\delta^{min}_{value} = \min_r(|P^{(r)}_1 - P^{(r)}_2| + |P^{(r)}_2 - P^{(r)}_3| + |P^{(r)}_3 - P^{(r)}_4|) \tag{3.2}$$

$$\delta^{min}_{label} = \operatorname*{argmin}_r(|P^{(r)}_1 - P^{(r)}_2| + |P^{(r)}_2 - P^{(r)}_3| + |P^{(r)}_3 - P^{(r)}_4|) \tag{3.3}$$

2. Median Motif Patterns and Labels:

$$\delta^{med}_{value} = \operatorname*{med}_r(|P^{(r)}_1 - P^{(r)}_2| + |P^{(r)}_2 - P^{(r)}_3| + |P^{(r)}_3 - P^{(r)}_4|) \tag{3.4}$$

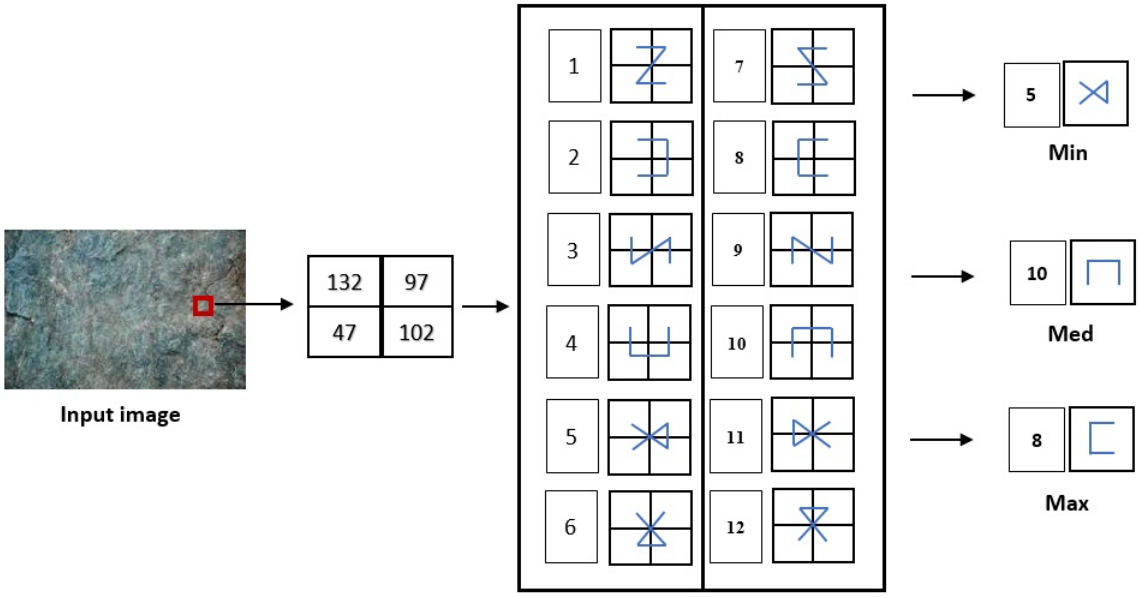


Figure 3.7: Illustration of motif labels spanning over 2×2 pixel neighborhood. Three moments are found from the 12 motif patterns extracted from each patch. each of these moments (Minumum, Median, and Maximum) is stored in a separate matrix along with their corresponding label. In total, we will have 6 matrices.

$$\delta_{label}^{med} = \underset{r}{argmed}(|P_1^{(r)} - P_2^{(r)}| + |P_2^{(r)} - P_3^{(r)}| + |P_3^{(r)} - P_4^{(r)}|) \tag{3.5}$$

3. Maximum Motif Patterns and Labels:

$$\delta_{value}^{max} = \underset{r}{max}(|P_1^{(r)} - P_2^{(r)}| + |P_2^{(r)} - P_3^{(r)}| + |P_3^{(r)} - P_4^{(r)}|) \tag{3.6}$$

$$\delta_{label}^{max} = \underset{r}{argmax}(|P_1^{(r)} - P_2^{(r)}| + |P_2^{(r)} - P_3^{(r)}| + |P_3^{(r)} - P_4^{(r)}|) \tag{3.7}$$

After creating the 3 ML matrices, we can find the histogram of each matrix and then combine the 3 histograms to get the final feature vector and use it to perform the classification task. Each of the 3 ML matrices will give 12 bins since we are extracting 12 motif patterns only. As a result of combining all of them we will gain 36 bins. As we can see in ML, we are combining three statistic orders instead of just minimizing the intensity variations among adjacent pixels. The advantage of using min, med, and max moments is to make the final histogram more robust and capable

of extracting complex texture features. Moreover, we will join each ML matrix with other information like mean and variance of the image to generate a more powerful descriptor. We can easily conclude that these motif peano scans have the same effect as the LBP descriptor by considering them as filters and can be convolved with the corresponding part of the small region to detect specific features as we can see in Figure 3.8 below.

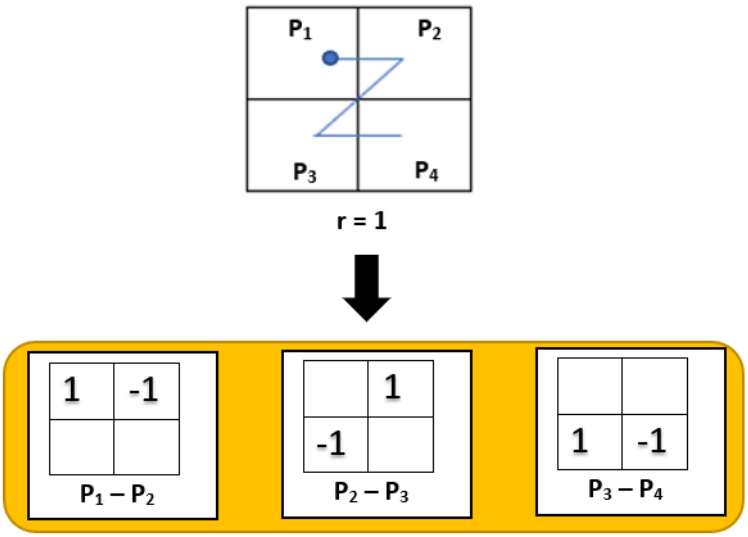


Figure 3.8: Convolutional implementation of Z pattern.

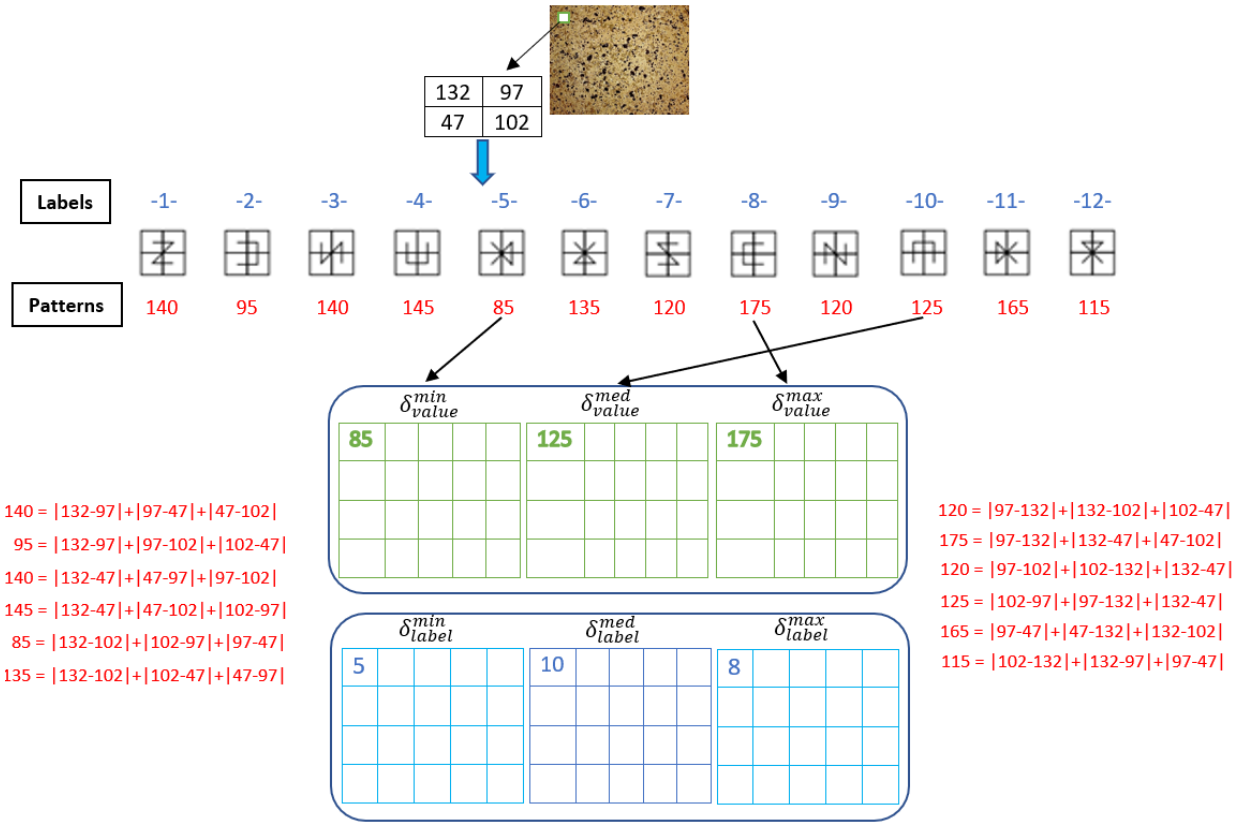


Figure 3.9: Simple example of how to extract the three motif labels and motif patterns from a 2×2 image patch. At the end of this process, we will generate six matrices, three for motif labels and another three for the corresponding motif patterns.

Figure 3.9 gives a more illustrative example with calculations of how we obtained these 6 matrices. We considered a small neighborhood patch of 2×2 and we applied algorithm 1 to find the six corresponding motif label and pattern matrices. It is worthy to mention that this new representation and use of motif labels has never been proposed before.

3.3.2 Joint Motif Labels (JML)

The motif labels descriptor generated using three statistc orders has the ability to capture texture information from the input image using 2 x 2 neighborhood scanning. However, to further enrich the descriptor and as the texture problem becomes difficult

especially with large scale datasets and big image sizes, more information must be extracted from the image and jointly added to that descriptor. The idea of joining features with the histogram distribution is not new. In [48, 3], Gue et al. and Hafiane et al. captured global information from the image using global thresholding and combined it with the magnitude of the local differences and the adaptive median binary pattern descriptor respectively. In our work, we capture two information from the original image: variance information and global thresholding.

For the variance information, we start capturing the local variance of each intensity pixel around a 3 x 3 neighborhood. This can be done by finding the standard deviation of each pixel around the 3 x 3 neighborhood and then square each value to get the corresponding variance. In this case, each pixel of the original input image will have a separate value that represent the variance around a 3 x 3 neighborhood. In order to capture the standard deviation of a group of values, we use the following equation:

$$S = \sqrt{\frac{\sum_{k=1}^{n}(x_k - \mu)^2}{n}} \tag{3.8}$$

In this case, the variance will be the square of S. Note, in the implementation, we used Matlab built in function *stdfilt* to find the standard deviation of each pixel in the image around a patch of 3 x 3 pixels. Later, we square each value to get the final variance matrix of all image pixels. After obtaining the variance matrix, we can threshold its values either by global variance of the image or by the mean of the values of that variance matrix (VAR). In our previous work [106], we used the mean value of the variance matrix instead of the global variance because thresholding against the global variance dropped the classification performance. In our work, we also use the same approach as before:

$$\gamma(i,j) = \begin{cases} 1, & \text{if } (i,j) \geq \mu \\ 0, & \text{otherwise} \end{cases} \tag{3.9}$$

where μ is the mean of the VAR matrix. γ generates two bins from this thresholding to indicate to which region each pixel belongs to.

For the global thresholding υ, thresholding was performed by comparing each intensity value of the input image against the mean of the intensities of the entire image. As we did with variance, global thresholding will also generate two bins depending whether each pixel is greater or equal to the mean intensities or not. These information now can be encoded with ML features as 3D joint histogram or even as 2D. Note, we will use the same algorithm in [3] to perform the joint process between the three matrices: ML, γ, and υ. Finally, the feature vector for each of the 3 ML matrices after joining both γ, and υ will be 48 bins. As a result, Joint Motif Labels descriptor will have 48 * 3 = 144 bins in total. More information can be found in our published paper [113].

3.4 Motif Patterns (MP) Encoded with RIC-LBP

The JML features computed before used only the labels of the 12 motif patterns extracted from traversing each pixel in the image around a 2 x 2 neighborhood. In addition, JML was used successfully as we mentioned to classify HEp-2 cell images and the performance was superior when combined with a robust descriptor like RIC-LBP. However, we did not make use of the patterns generated from this process. These patterns are very important since they represent the variation of intensity values of the traversed pixels. As discussed in section 3.3, when we computed the labels for the minimum, median, and maximum matrices, we also stored the corresponding patterns of these statistics orders. As a result, there will be three Motif Pattern (MP)

matrices representing the min, med, and max patterns resulting from traversing the 2 x 2 neighborhood.

Now, since these patterns represent intensity differences among pixels, we can consider them as an image transformation and use any texture descriptor to encode them. Since RIC-LBP proved to be efficient and robust descriptor, we decided to encode these three MP statistic orders with RIC-LBP. As we know, RIC-LBP generates 408 bins feature vector. Applying it to the three MP matrices will result in 1224 bins feature vector. In this work, we decided to use only the minimum MP matrix in order to reduce the feature vector size especially when we combine this descriptor with other descriptors. As a result, we get only 408 bins feature vector for the MP descriptor as illustrated in Figure 3.10 below.

Figure 3.10: Pipeline for the new MP_{RIC_LBP} calculations. After computing the three motif patterns (Min, Med, and Max), only the Min motif patterns is used to be encoded with RIC-LBP descriptor. This is because the Min moment represent the absolute difference between adjacent pixels in the 2×2 neighborhood and can be encoded with any local descriptor. The result of this new descriptor is 408 bins which will be combined with JML descriptor and used as features for the texture classification task.

In order to account for the **translational invariance** for both JML and MP descriptors, four feature vectors are extracted instead of one for each descriptor. Since image shifting can happen at any direction in real life, we need to extract our features from the original image, image shifted vertically, horizontally, and diagonally by one pixel. This method was used previously by Jhanwar et al. [1] and more details can be found in their paper. The result of applying translational invariance on both descriptors obviously will make the dimensionality bigger now. For JML, instead of having 144 features, now we have 144 * 4 = 576 features. For MP descriptor, we have

now 408 * 4 = 1632 features using only the minimum patterns. The performance of both descriptors will be evaluated in the next section along with the other texture descriptors described previously. Figure 3.11 below illustrates the effect of shifting one pixel on generating different motif patterns.

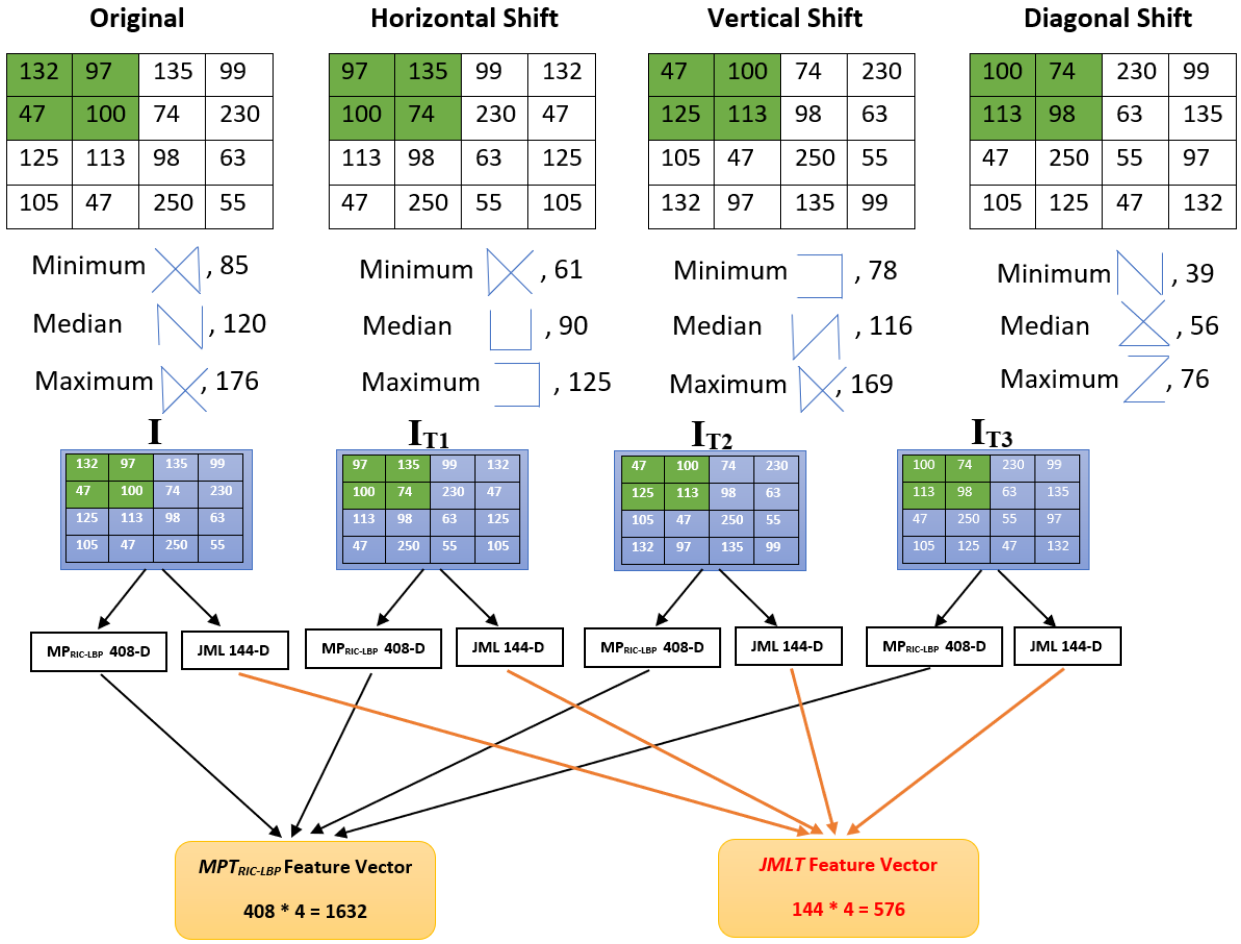


Figure 3.11: The effect of image shifting in motif pattern calculations. As we can see, any small shifting, even by one pixel, can result in a different motif patterns. To diminish this effect, we need to compute three other matrices representing the original image shifted by one pixel horizontally, vertically, and diagonally. At the end, we extract features from all these four images. MPT and JMLT refers to the Motif Patterns with Translation and Joint Motif Labels with Translation respectively.

3.5 Deep Features

We have also utilized deep learning in our work. As we detailed in the literature review chapter, there are different ways to use deep learning. It is possible to use end-to-end learning, in this case, we can use the softmax layer as a classifier and get the final accuracy. Or, it is also possible to extract features from different layers and use them with another classifier like RF. We found that the latter approach can generate better accuracy. Hence, we used these features in our framework. Figure 3.12 illustrates the difference between classical and traditional approaches for image classification.

3.5.1 VGG Network Features [4]

After the success of the AlexNet architecture in 2012, many attempts have been made to further extend that work to achieve superior performance. One of the attempts was the work of Simonyan et al. [4] in which they investigated the depth of convolutional neural networks by adding more convolutional layers. The result was a robust architecture that can significantly achieve high performance especially on large scale datasets.

The architecture of VGG networks are as shown in Figure 3.13. The input image size should be $224 \times 224 \times 3$. After that, the image will pass through a stack of convolutional layers. The convolution filters have a small receptive field of 3×3. The convolution stride is fixed to 1 pixel and the spatial padding of conv. layer input is 1 pixel for 3 convolution layers. Five maximum pooling layers are used for spatial pooling, however, not all the convolutioanl layers are followed by max-pooling which is performed over a 2×2 pixel window with stride of 2. Three Fully-Connected (FC) layers come after a stack of convolutional layers. The first two have 4096 channels each, the third one performs 1000-way ILSVRC classification and thus involves 1000

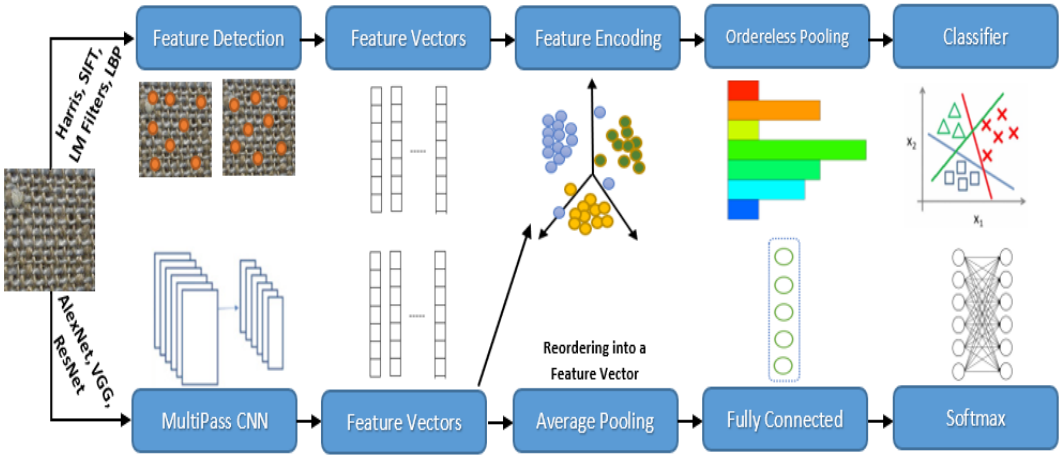


Figure 3.12: Classical vs traditional approaches for image classification. Deep learning layers replaced feature encoding with average pooling (reordering). In addition, instead of creating a final histogram of features with the traditional approaches, deep learning uses fully connected layers. Finally, softmax layer in deep learning replaces the classification stage of the traditional methods. Moreover, it is also possible to combine both features and use them with a classical classifier.

channels. The final layer in VGG is the softmax layer. In addition, All hidden layers are equipped with ReLU non-linearity. The VGG-19 network differs from the corresponding 16 one in that it has three more convolutional layer. Authors discovered that the classification error decreases with when the number of convolution layers increases.

In our work, we focused on VGG-10 since it showed better performance. We were successful to improve our texture classification pipeline that comprises of extracting multiple local descriptors by combining additional deep features extracted from 'fc7' layer of VGG-19 network.

3.6 Combining Local and Deep Features for Image Classification

Our final framework for texture classification involves using a combination of different features as we explained in the previous sections and shown in Figure 3.13. The deep

learning features extracted from 'fc7' layer showed excellent performance especially when Random Forests (RF) classifier is used in the classification stage. Furthermore, the classification performance was improved when we added our proposed set of local descriptors to these deep features. All features where combined in a late fusion mechanism and fed directly into RF classifier for training and testing. In the experimental results chapter, we will demonstrate the performance of our framework on the biomedical and the challenging texture datasets. The idea of combining multiple features is not new. Cimpoi et al. [24] used global features (like Fisher Vector (FV)) and deep features extracted from convolutional layers and Fully-Connected layers to perform classification on texture datasets. However, in our work, we used our proposed texture features along with the extracted deep features and showed that RF classifier can achieve superior performance with these features. The 4096 'fc7' features extracted from VGG-19 are considered as global features. We found that extracting these features after training the network properly and use a second step learning (using RF classifier) can improve the classification performance over the softmax accuracy. Our

More information on how we used our framework can be found in our published paper [114].

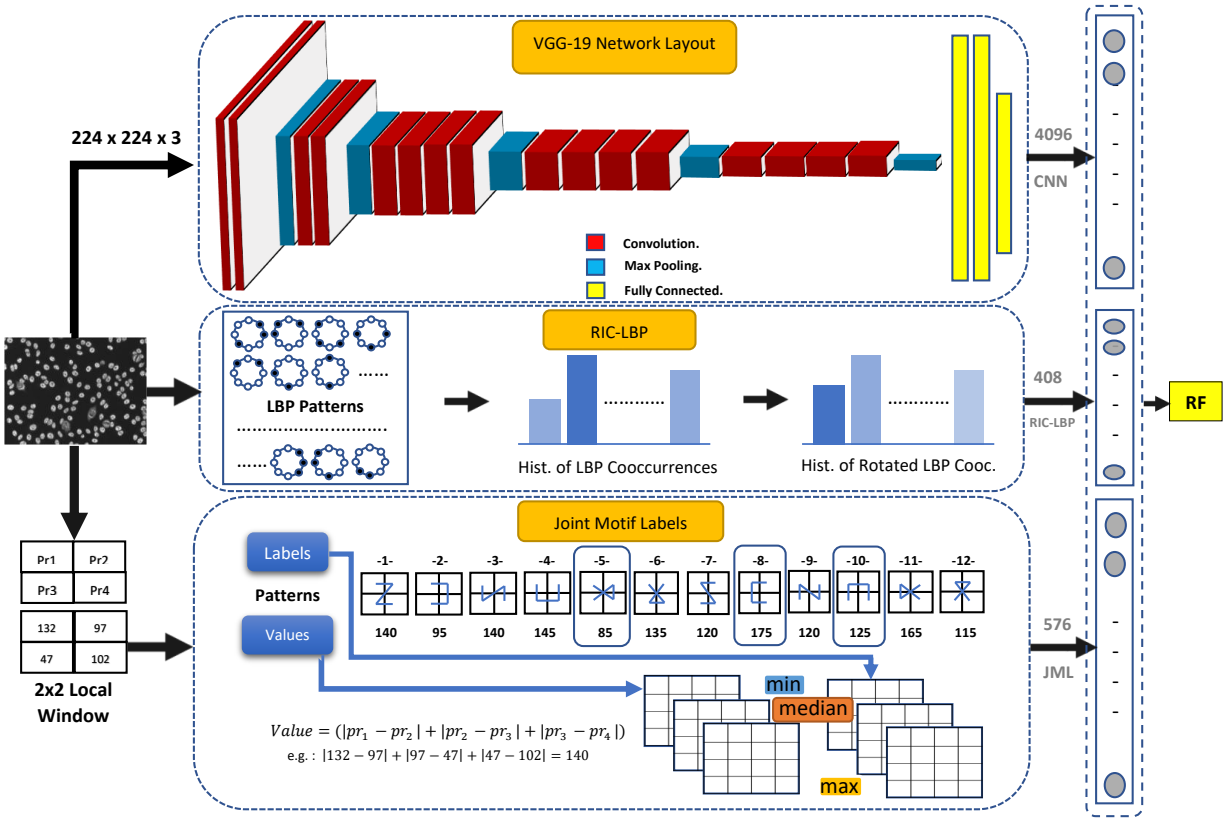


Figure 3.13: Overview of the proposed late fusion approach. Three types of deep and local features were extracted: CNN, RIC-LBP, and JML features. All features are concatenated and a Random Forests (RF) classier is applied to achieve high accuracy.

Chapter 4

Datasets

4.1 HEp-2 datasets

Two benchmark datasets were used for HEp-2 cell and specimen level classification. These datasets are referred to as Task 1 and Task 2 and were collected between 2011 and 2013 at the Sullivan Nicolaides pathology laboratory, Australia. For each task, a set of training images was provided to the contest participants. Submitted systems were then evaluated on a separate hidden test set which was privately maintained by the contest organizers and not released to the participants.

4.1.1 HEp-2 cell classification: Task 1

The Task 1 dataset consists of 68,429 images of individual cells extracted from 419 patient positive sera (approximately 100–200 cell images per patient serum) along with their binary segmentation masks. 13,596 images were available during training. The remaining 54,833 images were used for the hidden test set to evaluate performance of systems submitted to the contest. The specimens were automatically photographed using a monochrome high dynamic range cooled microscopy camera. Cell images are

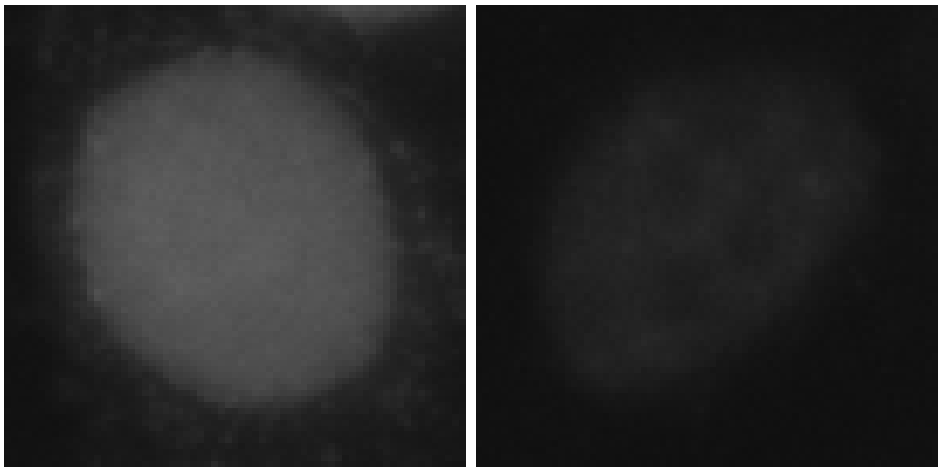
Class	Task 1	Task 2
Homogeneous	2494	212
Speckled	2831	208
Nucleolar	2598	200
Centromere	2741	204
Nuclear membrane	2208	84
Golgi	724	40
Mitotic Spindle	–	60

Table 4.1: Classes and number of images per class for both Task 1 and Task 2 training datasets. As we can see, both datasets consist of unbalanced number of images in each class. As a result, a class like Golgi can perform poorly compared to the other classes because fewer number of images are available for the training stage.

approximately 70×70 pixels in size. The dataset has six pattern classes: homogeneous, speckled, nucleolar, centromere, nuclear membrane, and golgi. Number of images per class for this dataset are shown in Table 4.1.

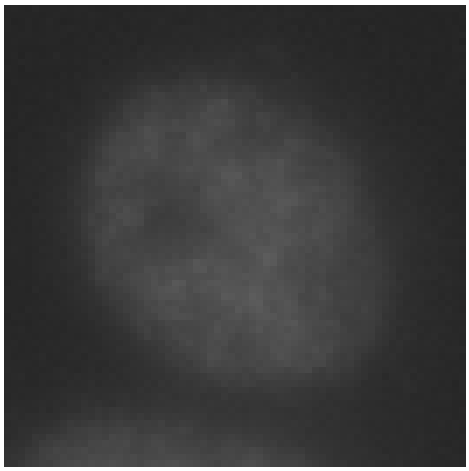
4.1.2 HEp-2 specimen classification: Task 2

The Task 2 dataset consists of uncompressed, monochromatic images of 1001 patient sera with positive ANA test. Each specimen was photographed at four different locations (four images per specimen). A total of 1008 images from 252 specimens were made available (approximately 25% of the data) while the remaining images were retained by the organizers for testing. Size of each image is 1388×1040 and cell masks were obtained based on an automatic segmentation for each image. The dataset has seven pattern classes: homogeneous, speckled, nucleolar, centromere, nuclear membrane, Golgi and mitotic spindle. Table 4.1 shows the number of images per class for this dataset. Images of Task 2 dataset are very large compared to Task 1. As a result, using multiresolution analysis to extract texture features and then combine these features will be very useful to improve the classification accuracy.

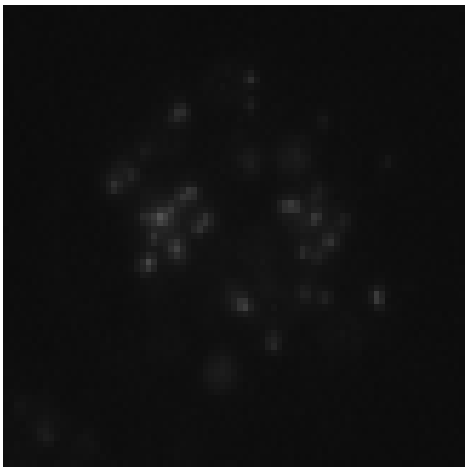


(a) Homogeneous

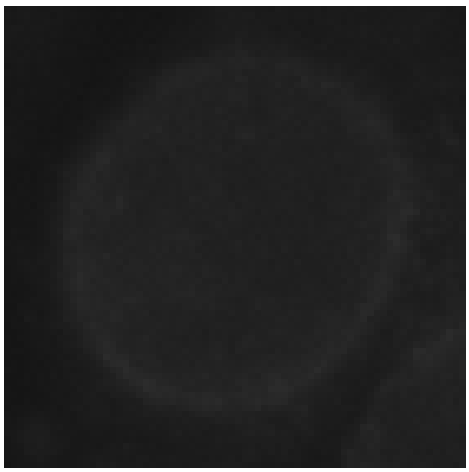
(b) Speckled



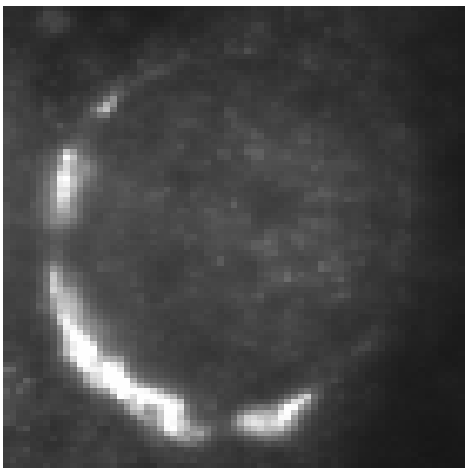
(c) Nucleolar



(d) Centromere

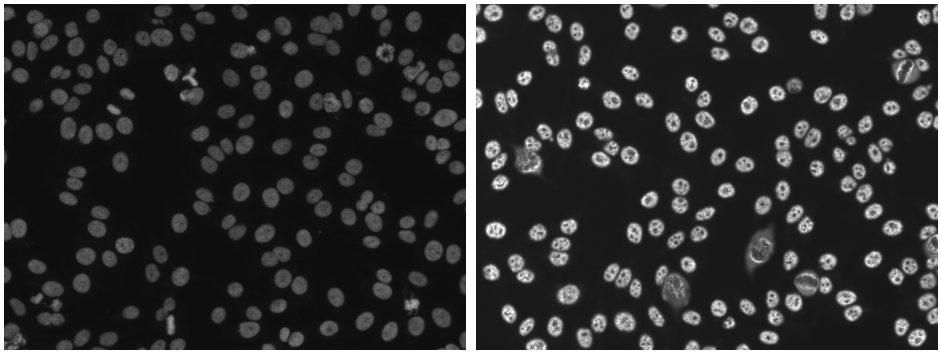


(e) Numem



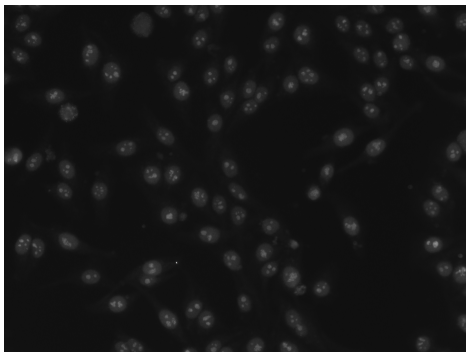
(f) Golgi

Figure 4.1: Sample cell images from each class. (a) Homogeneous. (b) Speckled. (c) Nucleolar. (d) Centromere. (e) Nuclear membrane. (f) Golgi.

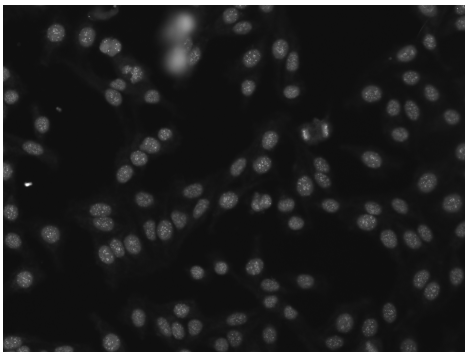


(a) Homogeneous

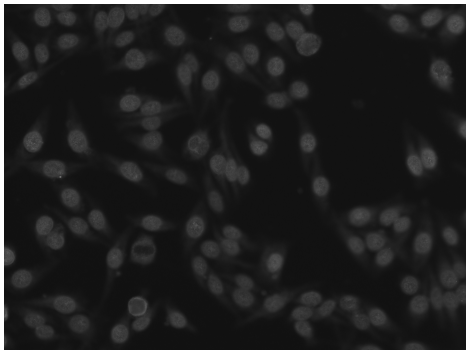
(b) Speckled



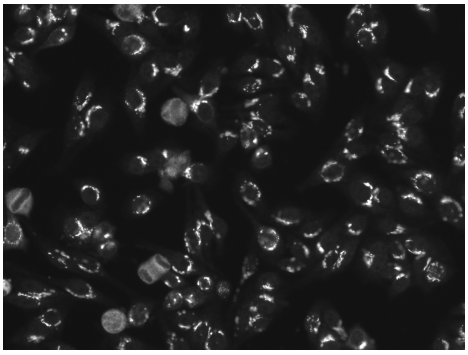
(c) Nucleolar



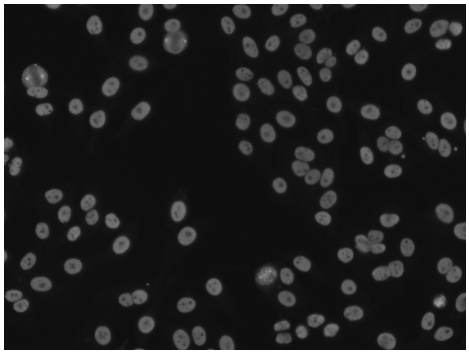
(d) Centromere



(e) Numem



(f) Golgi



(g) Mitsp

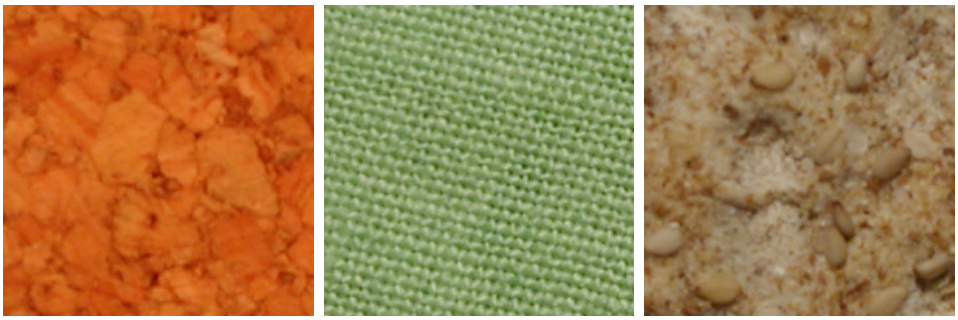
Figure 4.2: Sample Specimen images from each class. (a) Homogeneous. (b) Speckled. (c) Nucleolar. (d) Centromere. (e) Nuclear membrane. (f) Golgi. (g) Mitsp.

4.2 Standard texture datasets

4.2.1 KTH-TIPS-2a,b datasets

KTH-TIPS-2a and KTH-TIPS-2b were introduced in 2006 [115]. Images of both datasets are perfrect for scientific research since they are taken under varying variation, illumination, pose, and scale. Both datasets contain 11 classes. KTH-TIPS-2a images are captured at 9 different scales, 3 poses, and 4 different illumination conditions. In the experiments, we used 3 samples from each texture category for training and the remaining sample for testing and the mean class accuracy was found by taking the average over 4 runs. KTH-TIPS-2b images are also captured at different scales with 4 samples in each category with a total number of 4752 images. In the experiments, we used 3 samples for training and the last sample was used for testing and the mean class accuracy was also found over 4 runs. The size of images for both datasets is 200×200 pixels. The difficulty in both datasets also involves the possibility of having different surface coarseness or roughness in some categories which make them look different even if they are imaged with the same resolution. Moreover, the visual aspect of the texture also changes gradually with the resolution, which impacts the local structures.

(a) aluminium_foil (b) brown_bread (c) corduroy



(d) cork (e) cotton (f) cracker

(g) lettuce_leaf (h) linen (i) white_bread

(j) wood (k) wool

Figure 4.3: Sample images from 11 classes KTH dataset.

4.2.2 DTD dataset

The Describable Texture Dataset (DTD) was introduced by Cimpoi et al. [8] in 2014. Images of DTD were collected from the internet for the purposes of object categorization and recognition. The goal of DTD dataset is to support real-world applications with strong representation of texture properties in the given object. DTD uses 47 adjective English words adopted from Bhusan et al. work [116]. Instead of defining the texture problem as associating one attribute to each class, DTD associates multiple attributes for one class. For example, marble class can be veined, stratified, and cracked at the same time. DTD comprises of 47 classes, with 120 images per class and a total number of 5640 images. In the experiments, two third of the dataset will be used for training and one third for testing. Training and testing files are provided with the dataset, so, no random selection of images is made. Ten folds cross validation is applied during testing and the average mean class accuracy is taken as the final accuracy result.

Authors of this dataset also defined a gold standard representation that achieves state-of-the-art recognition on it and on other standard texture and material datasets. This involves performing thorough experiments using all types of features (local, global, and deep learning features). Experiments showed that global features and deep learning features are the top methods for classifying DTD classes. Moreover, combining global features with deep learning features will create more robust framework that further improves the classification accuracy.

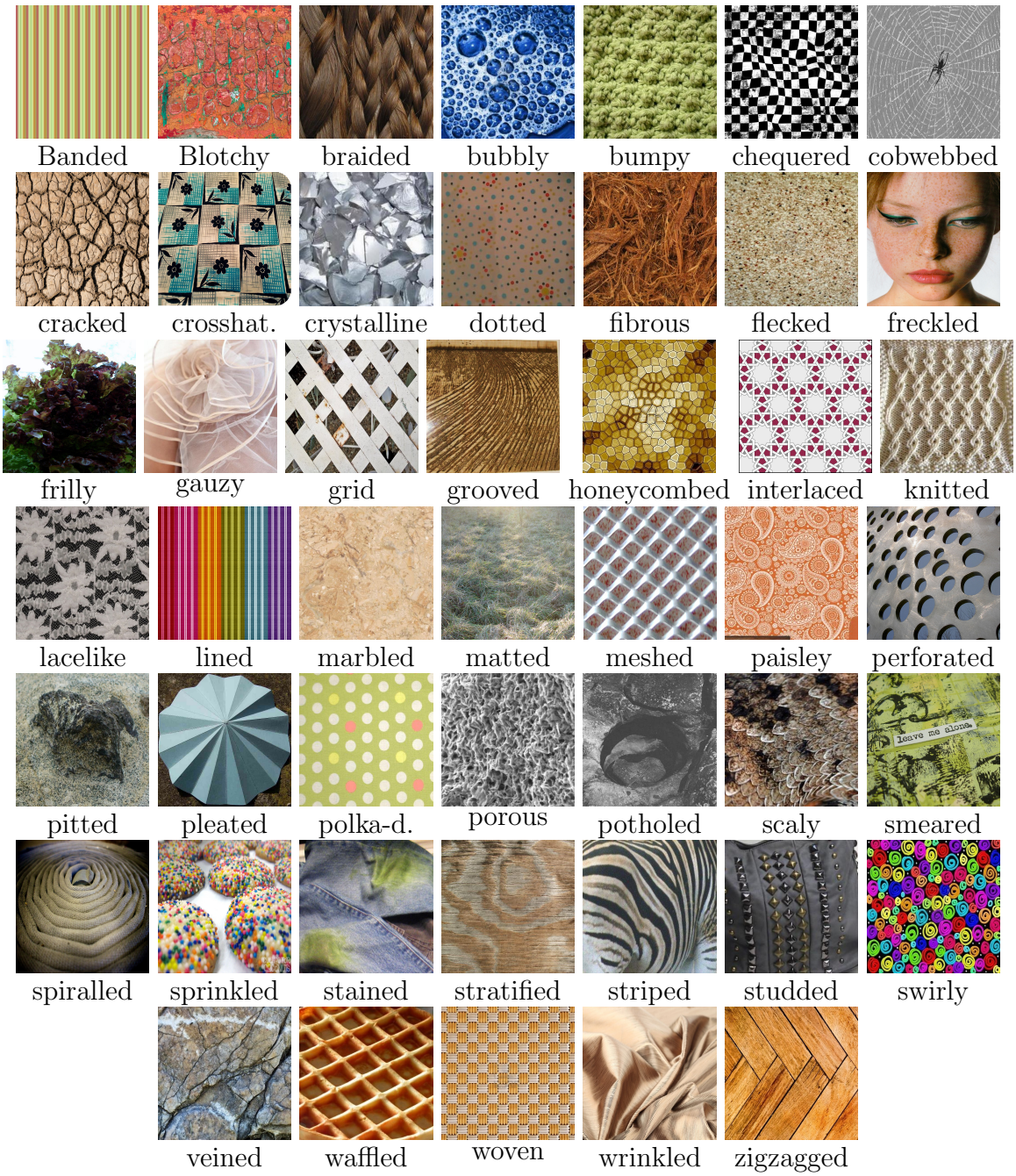


Figure 4.4: Sample images from DTD dataset. DTD contains 5640 images divided into 47 classes with each class has 120 image.

Chapter 5

Experimental Results

In this chapter, we test our feature extraction framework by using the standard HEp-2 and texture databases that were introduced in the previous chapter. The pipeline to perform experiments is given in Figure 5.1. Each database has specific number of classes with specific number of images in each class. According to the split procedure associated with each database, we start splitting the database into two sets: training and testing. Then, we extract local features for both sets using our framework. After that, we train a model using images from the training set. These models include Random Forests (RF), k-Nearest Neighbor (k-NN), and Support Vector Machines (SVM). Finally, we used these trained models to perform the classification task on the given test set of images and a final accuracy is computed. For the standard texture dabases, an associated split file is attached with each database that tells how many images to be used for training and for testing. Moreover, it also specifies the type of cross-validation metric to find the final accuracy. For example, DTD dataset is divided into 10 fold cross-validation. Which means, the pipeline for texture classification must be executed 10 times and the final accuracy is taken as the average of the 10 runs.

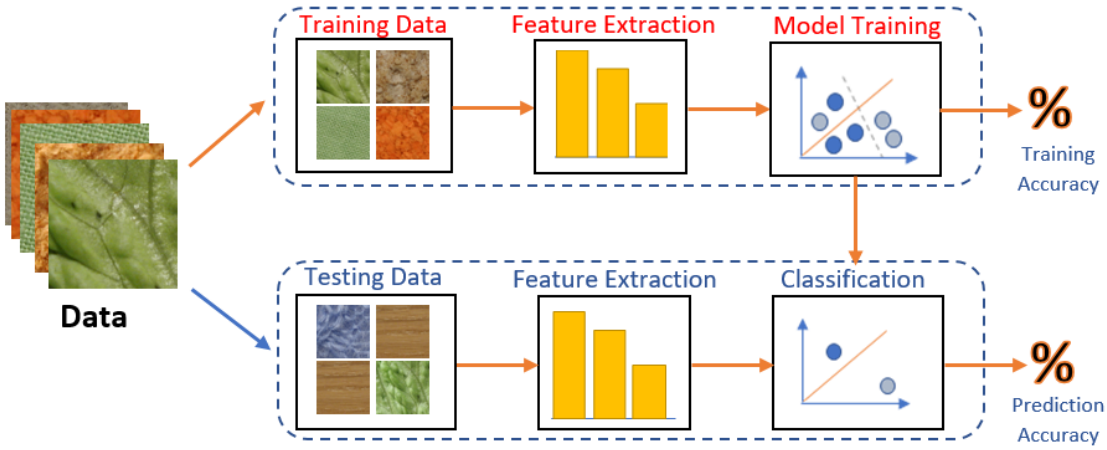


Figure 5.1: Image classification pipeline. First, the given pool of images of a specific application is divided into two sets: training and testing. Then, features are extracted for both sets using any given descriptor. After that, a model (like RFs, k-NN, or SVM) is trained on the given training set. Finally, a prediction is made based on the trained model and the test data and the final classification accuracy is generated.

However, for the HEp-2 databases (cells and specimen), since they are the training datasets, no protocol is associated with them to detail how many images to be used for the training and the testing phases. In this case, we followed the procedure that was used by most of the state-of-the-art methods which run experiments on these datasets. This involves dividing the datasets into 5 folds. In this case, 80% of the images will be used for training and 20% will be used for testing.

The evaluation metric used in our experiments is called the Mean Class Accuracy (MCA). MCA is computed as follows:

$$MCA = \frac{1}{N}\sum_{n=1}^{N} CCR_n \tag{5.1}$$

where CCR_n is the correct classification rate for class n. N is the total number of classes.

5.1 Classifiers

5.1.1 k-NN Classifier

k-Nearest Neighbors classifier is a non-parametric method used for classification or regression to solve pattern recognition related problems. In both cases, the input consists of the k closest training examples in the feature space. The output depends on whether k-NN is used for classification or regression [117]. In k-NN classification, the output is a class membership. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbor. The best choice of k depends upon the data, generally, larger values of k reduces effect of the noise on the classification, but the boundaries between classes will be less distinct.

5.1.2 RF Classifier

Random Forests (RFs) are an ensemble learning method for classification and regression which operate by constructing multiple decision trees at training time and outputting the class that is the mode of the classes or mean prediction in the case of the regression task [118]. Using the random subspace method, the first algorithm for random decision was created by Tin Kam Ho. After that, an extension of Ho's algorithm was developed by Breiman et al. [119] which involves combining Breiman's "bagging" idea and random selection of features, introduced first by Ho et al. and resulted in the Random Forests (RFs). The training algorithm for random forests applies the general technique of bootstrap aggregating, or bagging, to tree learners. The number of trees is a free parameter. Typically, a few hundred to several thousand trees are used, depending on the size and nature of the training set.

5.1.3 SVM Classifier

Support Vector Machines (SVMs) is considered as one of the powerful tools in machine learning [120]. SVMs are are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier. In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces. Various real world problems can be solved with SVM like text categorization, image classification and segmentation, and hand-written recognition.

5.2 Experiments on HEp-2 Databases using Local Descriptors

Two datasets were used in the experiments represent cell and specimen images. We used two benchmark classifiers in these experiments: k-NN and RF. Both classifiers showed high accuracy and promising results using our framework for feature extraction. When we started developing our framework, we combined only two features: RIC-LBP and JML. Moreover, JML descriptor did not include the translational invariance property since these cell and specimen images contain no rotation or scale changes. As a result, the combined feature vector of both descriptors equals: 552 bins since RIC-LBP generates 408 bins and JML with no translational invariance generates only 144 bins.

5.2.1 Experiments on HEp-2 Cell Level Classification

Task 1 dataset consists of six classes and a varying number of images per class. Golgi class has the fewest number of images among all classes. In the experiments, we used five-fold cross validation across the dataset and images were selected randomly in each fold for training and testing. Final MCA was found and our results as in Table 5.1 outperforms the state-of-the-art LBP variation (RIC-LBP) by almost 2% using the k-NN classifier.

Descriptor	Size	Classifier	Accuracy	MCA
RIC-LBP, JML	552	k-NN	94.26	93.16
RIC-LBP, JML	552	RF	93.34	91.02
RIC-LBP, MCL-Min/Med/Max	840	k-NN	92.75	91.69
RIC-LBP, MCL-Min/Med/Max	840	RF	90.70	88.32
RIC-LBP, ML(Min/Med/Max)-GVAR	552	k-NN	92.57	91.36
RIC-LBP, ML(Min/Med/Max)-GVAR	552	RF	92.56	90.38

Table 5.1: Cell level staining pattern classification (Task 1) with motif texture pattern features. Here we show the overall accuracy and mean class accuracy (MCA) for two different classifiers, k-nearest neighbors (k-NN), and random forest (RF).

In comparison with state-of-the-art, Mannivannan et al. [131] achieved the best accuracy of 95.2%. However, they used more complicated set of features which involved local and global features. The accuracy that we generated is still comparable using our set of local features. Table 5.2 shows the confusion matrix of our best result. As we can see, only Golgi class generates the lowest accuracy among the 6 classes. The reason behind that is because there are fewer number of images in this class compared to the other classes.

5.2.2 Experiments on HEp-2 Specimen Level Classification

In Task 2, there are seven classes and a total number 1008 images distributed unequally among these classes. Table 5.3 shows the results of performing experiments using both JML and RIC-LBP descriptors with two classifiers: k-NN and RF. We

	Homogeneous	Speckled	Nucleolar	Centromere	Golgi	Numem.
Homogeneous	**95**	3	1	0	1	0
Speckled	2	**92**	4	1	1	0
Nucleolar	1	1	**95**	2	0	1
Centromere	0	1	1	**97**	0	0
Golgi	4	1	9	2	**85**	0
Numem.	3	2	1	0	0	**95**

Table 5.2: Confusion matrix for the cell level staining pattern classification (Task 1) with RIC-LBP, ML-LVAR texture features and k-NN classifier (with rounded percentages). The overall average accuracy is 94.26%. The class with the lowest accuracy is Golgi, because it has fewer images compared to other classes.

also perform the experiments on RF with various number of trees. The performance can be increased as demonstrated by using multiresolution analysis of the original image. As illustrated in Figure 5.2, L1 means the original image is subsampled and the framework for texture classification is applied to both the original and the subsampled images.

Descriptor	Size	Classifier	Accuracy	MCA
RIC-LBP, JML	552	k-NN	73.47	70.09
RIC-LBP,	408	k-NN	67.32	64.79
JML	144	k-NN	66.14	60.96
RIC-LBP, JML	552	RF-500	79.14	72.69
RIC-LBP, JML	552	RF-250	79.26	73.24
RIC-LBP, JML	552	RF-100	79.94	73.92
RIC-LBP,	408	RF-500	75.56	68.51
RIC-LBP,	408	RF-250	76.46	69.55
RIC-LBP,	408	RF-100	76.35	68.49
JML	144	RF-500	72.55	64.12
JML	144	RF-250	73.7	65.46
JML	144	RF-100	73.24	64.72
RIC-LBP/L1, JML/L1	1104	RF-500	82.44	78.33
RIC-LBP/L1, JML/L1	1104	RF-250	82.57	78.6
RIC-LBP/L1, JML/L1	1104	RF-100	81.8	77.92
RIC-LBP/L1,	816	RF-500	78.46	72.64
RIC-LBP/L1,	816	RF-250	78.36	72.08
RIC-LBP/L1,	816	RF-100	77.64	72.27
JML/L1	288	RF-500	76.5	69.54
JML/L1	288	RF-250	76.91	69.88
JML/L1	288	RF-100	76.8	69.43
RIC-LBP/L2, JML/L2	1656	RF-500	88.71	84.85
RIC-LBP/L2, JML/L2	1656	RF-250	88.33	84.84

Table 5.3: Task 2 results using RIC-LBP and JML descriptors with k-NN and RF classifiers. L1/L2 means Level one and Level two respectively. This is because we subsample the original image twice and extract the features of both the original image and the subsampled ones as illustrated in Figure 5.2

In comparison with state-of-the-art, Mannivannan et al. [131] achieve an accuracy of 89.9% on this dataset. As we mentioned above, they use more robust features. However, we will show later how we can beat their accuracy on this dataset by considering the deep features to our framework.

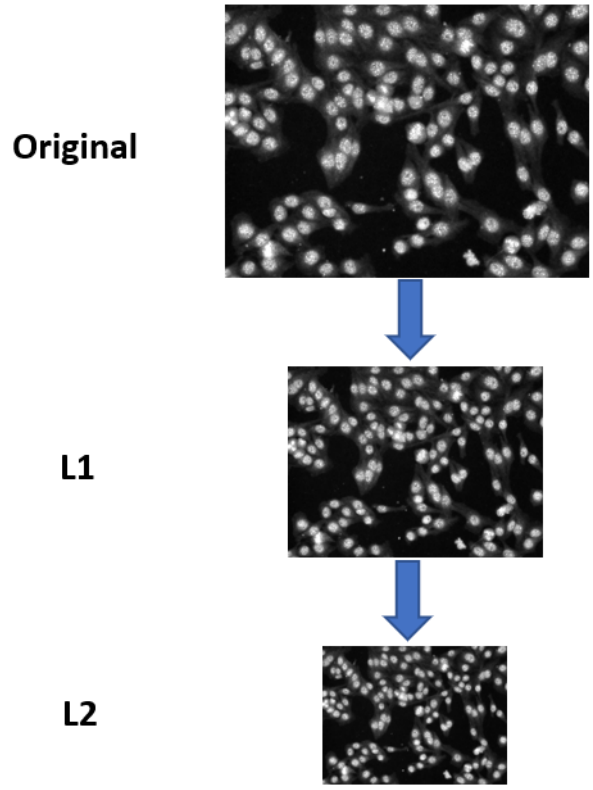


Figure 5.2: Illustration of the subsampling technique used in Task 2 specimen dataset. The original image is resized twice and the features are extracted from all three images.

5.3 Experiments on HEp-2 Databases using Local and Deep Features

The following experiments focus on classifying HEp-2 specimen cells using our proposed set of local descriptors and deep learning features. For both HEp-2 databases, RF classifier was used with 1000 trees. We extracted 4096 bins from 'fc7' layer of VGG-19 architecture. We have also trained our network using the Caffe toolbox [130] on a single GPU of 6 Gigabyte of cache memory. The minimum batch size used was set to 40 samples and the learning rate was fixed to 0.001. The total number of epochs in each experiment was fixed to 10.

5.3.1 Classifying HEp-2 Cell Images using Deep Learning

We applied five-fold cross validation on the HEp-2 cell images which involves six classes. Since Golgi class contains fewer images than the rest of the classes, we augmented images of that class by performing image rotation using three degrees $(90^0, 180^0, 270^0)$ in order to make images of that class in level with other classes. The results of applying net-to-net accuracy was 95.99%, which is very high. The results of applying our proposed features are as in the table below:

Descriptor	Desc. Size	Classifier	MCA
VGG-19	4096	RF - 1000	**97.42**
RIC-LBP	408	RF - 1000	88.59
JAMBP	320	RF - 1000	77.47
JML	576	RF - 1000	84.45
MP	1632	RF - 1000	76.01
VGG-19 + RIC-LBP	4504	RF - 1000	97.4
VGG-19 + RIC-LBP + JAMBP	4824	RF - 1000	97.37
VGG-19 + RIC-LBP + JAMBP + JML	5400	RF - 1000	97.39
VGG-19 + RIC-LBP + JAMBP + JML + MP	7032	RF - 1000	**97.43**

Table 5.4: Results of Applying our Proposed Descriptors. As we can see, the deep features already generated very high result. As a result, combining deep and local features did not improve the overall performance.

From Table 5.4, we can see the performance of deep features is very good. The 92.42% could not be improved when we included our set of features. In addition, local features also achieved good results. This is because we have enough data for training and for testing with relatively small number of classes.

We have compared the performance of our approach with state-of-the-art methods produced by Mannivannan et al. [131] in which a mean class accuracy of 95.2% was achieved using Root-SIFT features and multi-resolution LBPs from HEp-2 cells with ensembles of SVMs for the classification phase. They showed high accuracy using two-fold cross validation.

5.3.2 Classifying HEp-2 Specimen Images using Deep Learning

In this database, we have also used a five-fold cross validation on seven specimen classes. Since, some of the classes also have fewer number of images compared to other classes, we have applied image augmentation by rotating images by ($90^0 and 180^0$) degrees. The results of applying net-to-net accuracy was 78.85%, which was further improved to 83.04% by sampling more batches from each image and feed the network with more samples than just re-sizing the original image to the required VGG-19 input image size which is restricted to 224×224. The results of applying our proposed features are as in the table below:

Table 5.5 shows the results of applying our framework with 1000 Random Forests (RF) classifier after carrying out five-fold cross validation experiments. First, deep learning features (4096 bins) generated a very good result of 90.81% MCA. This result was further improved by 1.3% after combining both local descriptors: RIC-LBP and JML with deep learning features.

Table 5.6 shows a comparison between our approach and the previously state-of-the-art techniques. Mannivannan et al. [131] extracted a combination of Root-SIFT features and multi-resolution LBPs from HEp-2 image cells with ensembles of SVMs for the classification phase. They achieved high accuracy of (89.93%) and were the winners of I3A 2014 competition. Li et al. [110] employed a fully convolutional

Descriptor	Size	RF
VGG-19	4096	90.81
RIC-LBP	408	70.14
JML	576	66.35
VGG-19+RIC-LBP	4504	91.32
VGG-19+RIC-LBP+JML	5080	**92.11**

Table 5.5: Results comparing late fusion of deep and local features with our other approaches using RF classifier.

Method	MCA
Liu et al. cited in [131]	86.10
Gragnaniello et al. [105]	86.77
Mannivannan et al. [131]	89.93
Li et al. [110]	90.89
Ours (early fusion)	81.90
Ours (late fusion)	**92.11**

Table 5.6: Comparison of our approach with state-of-the-art methods. We outperformed all the existing methods in the literature for classifying the 7 specimen classes by more than 1%. The powerful performance of our approach also can be attributed to the use of 1000 trees RF classifier which which outperformed SVM classifier used in majority of other methods.

network and used the VGG-16 softmax layer to achieve an accuracy of 90.89%. Our framework of combining local and deep learning features slightly outperforms these state-of-the-art methods.

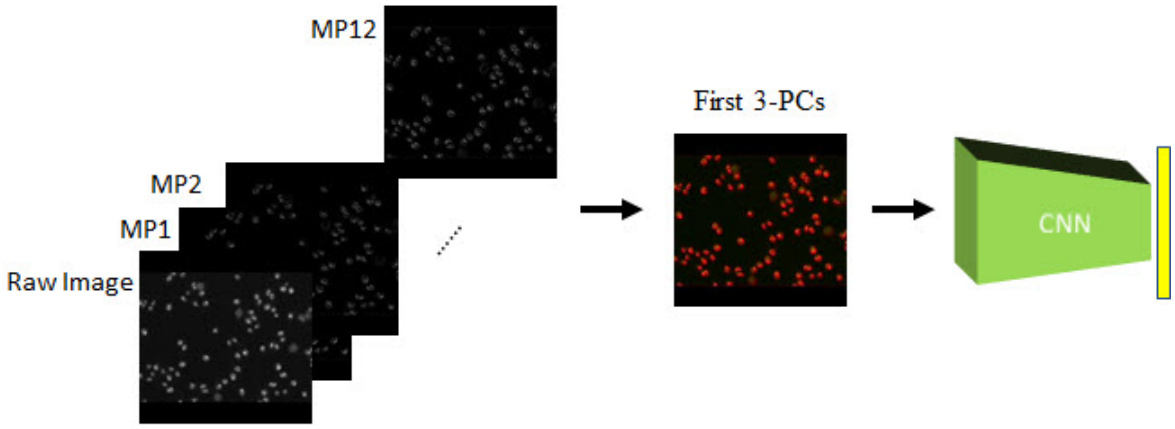


Figure 5.3: Early fusion mechanism used in our experiments. The 12 motif patterns produced in our approach are combined with the gray-scal specimen image. Then, we apply PCA to the 13-channel matrix and choose the first 3 Principle Components (PCs) only and feed the resultant 3-channel image to the CNN.

In addition to late fusion mechanism, we have tried early fusion mechanism with Principle Component Analysis (PCA) as illustrated in Figure 5.3. Since we have the original gray-scale image and the 12 motif patterns results from using our approach, we decided to combine all of them in one matrix and apply PCA. Then, we took

the first three components and fed the 3 channels resultant image to the VGG-19 network. The performance we obtained from end-to-end learning was only 81.9% which is significantly lower than using the late fusion mechanism.

5.4 Experiments on Standard Texture Databases using Local Descriptors

In this section, we also demonstrate the performance of our approach for texture recognition using three widely used datasets: KTH-TIPS-2a, KTH-TIPS-2b, and DTD [115, 8]. Two classifiers were used in the experiments: k nearest neighbor (kNN) and Support Vecotr Machine (SVM). kNN is considered as the widely used classifier for the texture classification problem by considering subset of dataset images for the training phase and the other subset for the testing phase. Superior classification performance is achieved using the chi-square distance χ^2 with kNN as follows:

$$\chi^2(x, y) = \frac{1}{2} \sum_i \frac{(x_i - y_i)^2}{x_i + y_i} \tag{5.2}$$

x and y represent feature vectors. We also considered 1 and 3 kNNs for all experiments.

For the SVM classifier, Cimpoi et al. [8] performed experiments using different SVM kernels and showed that the performance increases with Chi-Square kernel. In our work, we use SVM with two kernels: Linear $K(x, y) =< x, y >$ and Additive$-\chi^2 \sum_{i=1}^{d} x_i y_i/(x_i + y_i)$ where x and y are two feature vectors.

The experiments are divided into two main parts. The first part deals with KTH-TIPS-2(a,b) datasets using the standard testing and evaluation. The second part deals with the DTD dataset, with experiments performed on predefined training and testing sets. Gray-scale images were used among all experiments. In the following, JML represents the proposed Joint Motif Labels descriptor, while $MP_{RIC-LBP}$ represents

the proposed Motif Patterns descriptor encoded with RIC-LBP descriptor. In both descriptors, translational invariance is introduced.

5.4.1 Experiments on KTH-TIPS-2a,b

As we discussed in the previous chapter, both KTH-TIPS-2a,b databases are challenging ones because of the variations of illumination, pose, and scale. Both datasets contain 11 classes. KTH-TIPS-2a images are captured at 9 different scales, 3 poses, and 4 different illumination conditions. In the experiments, we used 3 samples from each texture category for training and the remaining sample for testing and the mean class accuracy was found by taking the average over 4 runs. KTH-TIPS-2b images are also captured at different scales with 4 samples in each category with a total number of 4752 images. In the experiments, we used 3 samples for testing and the last sample was used for training and the mean class accuracy was also found over 4 runs. Results on both datasets using our approach can be found in Tables 5.7 and 5.8. As we can see, the best results are achieved when we combine the four descriptors together. The highest accuracy we get for KTH-TIPS-2a is 79.2% and for KTH-TIPS-2b is 67.2%.

		k-NN		SVM	
Desc.	Size	k=1	k=3	Linear	additive$-\chi^2$
RIC-LBP	408	74.2	72.6	75.0	75.4
JAMBP	320	71.1	72.6	70.7	72.7
JML	576	65.9	68.0	67.1	67.3
MP	1632	65.6	67.3	69.5	68.8
RIC-LBP+JAMBP	728	76.0	75.2	75.5	76.7
RIC-LBP+JAMBP+JML	1304	76.1	74.8	76.9	77.2
RIC-LBP+JAMBP+JML+MP	2936	78.0	77.4	79.1	**79.2**

Table 5.7: Results on KTH-TIPS-2a dataset using our approach. As we can see from these results, SVM classifier performs better than k-NN. In addition, combining multiple descriptors can improve the performance. The accuracy with RIC-LBP using SVM is 75.4%. After combining four local descriptors, the accuracy was improved by approximately 4%.

		k-NN		SVM	
Desc.	Size	k=1	k=3	Linear	additive$-\chi^2$
RIC-LBP	408	58.7	59.6	62.7	62.6
JAMBP	320	59.6	60.5	57.8	61
JML	576	53	54.9	52.7	54.9
MP	1632	53.3	54.7	58.7	58.6
RIC-LBP+JAMBP	728	63.6	63.2	62.4	64.3
RIC-LBP+JAMBP+JML	1304	64.3	63.4	63.5	66
RIC-LBP+JAMBP+JML+MP	2936	66	65.3	65.6	**67.2**

Table 5.8: Results on KTH-TIPS-2b dataset using our approach. The SVM classifier beats the k-NN classifier by more than 2%. In addition, combining multiple local descriptors proved to improve the classification performance over using a single descriptor. We obtained an increase of 5% when we combined the four local descriptors.

5.4.2 Experiments on DTD Dataset

The Describable Texture Dataset (DTD) was introduced by Cimpoi et al. [8]. Images of DTD were collected from the internet using 47 adjective English words. Instead of defining the texture problem as associating one attribute to each class, DTD associates multiple attributes for one class. For example, marble class can be *veined*, *stratified* and *cracked* at the same time. DTD comprises of 47 classes, with 120 images per class and a total number of 5640 images. In the experiments, two third of the dataset will be used for training and one third for testing. Training and testing files are provided with the dataset, so, no random selection of images is made. Using our approach as can be seen in Table 5.9 we achieve a high result of 43.6%.

5.4.3 Experiments on KTH-TIPS-2a, b with Deep Learning

In the previous section, we showed the performance of our proposed set of local features on the standard texture datasets. It is clear that combining multiple local descriptors can result in a better performance compared to using only a single local descriptor. In this section, we demonstrate the performance of using deep learning classification and deep features in addition to our local descriptors. The mechanism

		k-NN		SVM	
Desc.	Size	k=1	k=3	Linear	additive$-\chi^2$
RIC-LBP	408	22.4	33.3	37.07±0.99	35.9±1.29
JAMBP	320	18.6	27.8	22.8±0.77	24.6±2.75
JML	576	18.5	29.3	21.0±0.77	23.9±0.6
MP	1632	16.4	26.8	31.1±1.23	32±0.91
RIC-LBP+JAMBP	728	21.9	32.7	37.0±1.11	38.4±1.33
RIC-LBP+JAMBP+JML	1304	24.0	35.4	38.3±1.05	40.1±0.55
RIC-LBP+JAMBP+JML+MP	2936	23.2	34.0	40.5±1.01	**43.6**±0.72

Table 5.9: Results on DTD dataset using our approach. Since the complexity of images in each class in this dataset is very high, local features can only achieve low accuracy. Starting from RIC-LBP, we obtain 35.9% using the SVM classifier with 10 fold-cross validation. After combining our proposed set of features, we improve the accuracy by more than 7%.

of fusing deep and local features is still the same. Hence, we can simply concatenate all features and use the final feature vector in the classification stage.

KTH-TIPS-2b represents a big challenge in terms of acquiring high classification accuracy. Figure 5.4 illustrates 3 challenging classes of the dataset. As we can see, these three classes: Corduroy, Linen, and Wool have images that are very much different in color, shape, and texture. This intra-class variation among images that belong to the same class makes it very difficult for any classifier to distinguish the correct class for each image under testing. Moreover, we should only use 25% of the dataset images for training, hence, we have only fewer number of images for training and many images for testing.

Corduroy Linen Wool

Figure 5.4: The three challenging KTH-TIPS-2b classes. We can cleary see that images of these classes differ in shap, color, and texture making it very difficult for the classifier to generate high accuracy for those classes.

The first experiment we performed on KTH-TIPS-2a was using deep learning and extracting features from 'fc7 layer. Then, after we do transfer learning, we classify these features with SVM classifier. The accuracy was 99.6% after performing four-fold cross validation. The reason behind this high accuracy is because the KTH-TIPS-2a dataset is easier in terms of image variations than the corresponding KTH-TIPS-2b. More over, we have less images compared to KTH-TIPS-2b and the standard protocol for splitting the dataset is different in both datasets. In KTH-TIPS-2b, we have to use one quarter of the dataset for training and the rest for testing. While in KTH-TIPS-2a, the opposite is applied.

For the KTH-TIPS-2b, as we mentioned above the data split puts fewer images for training and more images for testing. In theory, we can always conclude that the accuracy will not be as high as KTH-TIPS-2a since we do not provide the classifier with adequate number of images in the training stage. Hence, we need to perform augmentation to compensate for that. In the following experiments, we used different sampling and augmentation techniques in order to increase the number of images for the training stage.

- **Experiment 1**: In this experiment, we augmented the training images of KTH-TIPS-2b dataset by resizing the original image into 16 different images. First, the original row and column size of the image is divided by 4, then we resize the resultant dimension to 224×224 which are the input size of VGG-19 architecture in deep learning. Figure 5.5 illustrates this augmentation process. As a result, we managed to obtain 19,008 images from this augmentation process. Then, we used deep learning by training the VGG-19 network and get the end-to-end accuracy and the accuracy of extracting the 'fc7' features and performing RF-1000 classifier on these features. Results of this experiment are as listed in Table 5.10.

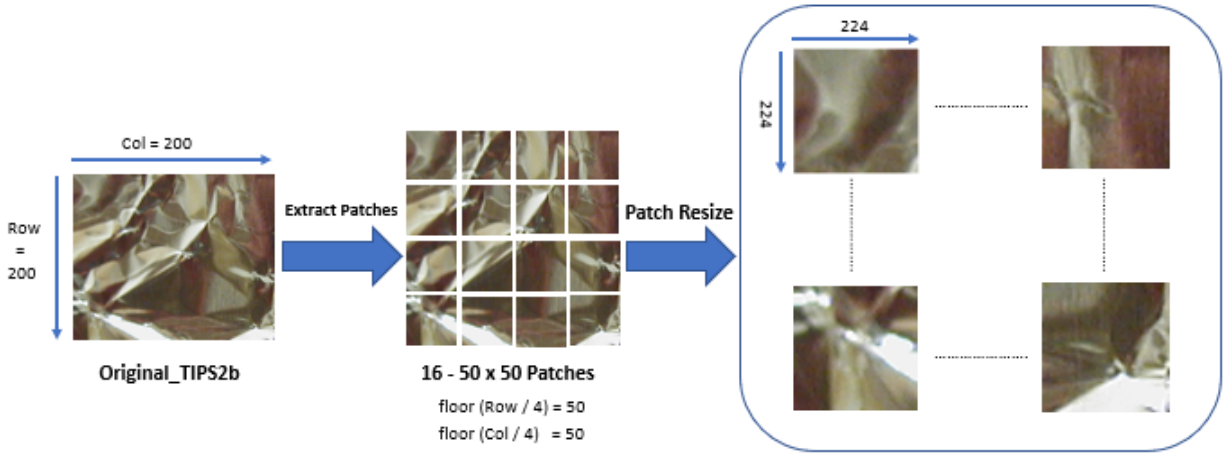


Figure 5.5: Augmentation of the original KTH-TIPS-2b image (Aluminium Foil). The original row and column size of the image is divided by 4, then we resize the resultant dimension to 224×224 in order to be used for the deep learning training and testing.

Descriptor	MCA
VGG-19	62.4
RIC-LBP	48.1
JAMBP	50.1
JML	41.6
MP	44.6
VGG-19 + RIC-LBP	62.6
VGG-19 + RIC-LBP + JAMBP	63.9
VGG-19 + RIC-LBP + JAMBP + JML	64.6
VGG-19 + RIC-LBP + JAMBP + JML + MP	**64.6**

Table 5.10: Results of applying our framework on the augmented KTH-TIPS-2b dataset using RF classifier with 1000 trees. As we can see using VGG-19 features alone we can get 62.4% accuracy, but, when we combine all the deep and local features, we can improve the accuracy to 64.6%.

During training the VGG-19 model, a MinBatchSize of 40 was used along with a fixed learning rate of 0.001 and the total number of Epochs was 10. The end-to-end accuracy of VGG-19 was only 51.6%. As we can see from these results, the feature extraction and classification results improved the performance massively. Moreover, combining the deep and the local features together also

helped improving the overall accuracy. Finally, we can see that this accuracy is still below the accuracy obtained by using the local features alone. This is due to the fact that the augmentation procedure used involves losing data from the original image. As a result, the network is only provided with part of the original object to be trained with.

- **Experiment 2**:

  As we mentioned in the previous experiment, the augmentation technique suffered from losing important image features. In this experiment, we used more powerful augmentation technique which involves rotation, replication, and translation. First, we create a large image of 600×600 in rows and columns. Then, we replicate the original image inside this large image as illustrated in Figure 5.6. We select 16 center points inside the large image in order to extract 16 (224×224) patches to be used in our training with deep learning. Finally, each extracted patch will be rotated by 90°, 180°, and 270°. In total, we extract 76,032 images from the original 1188 images provided for training. That means, each image will produce 64 patches for the training stage. We also used the RF classifier with 1000 trees after extracting the features from the 'fc7' layer of the VGG-19 architecture. Results of experiment 2 are as listed in Table 5.11.

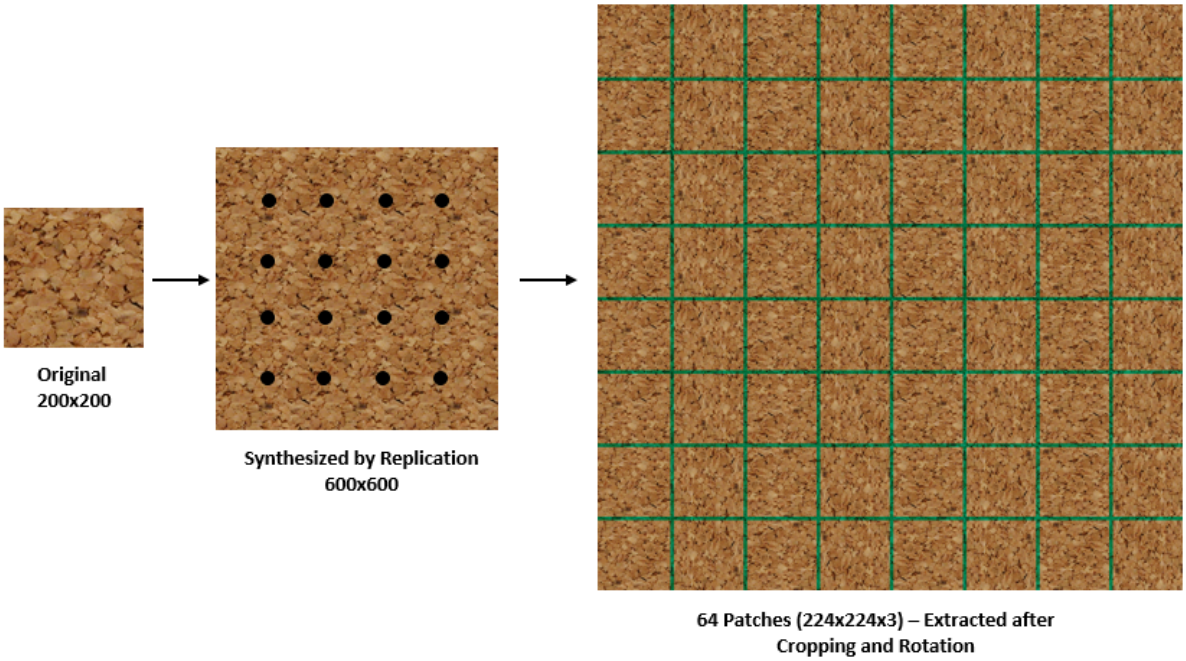


Figure 5.6: Different augmentation technique applied to the original KTH-TIPS-2b dataset. We synthesize a 600×600 large image and align the original image in it. The black dots represent the centers to extract 224×224 patches. After that we apply rotation to each patch. At the end, we obtain 64 patches from each image. Hence, we increase the 1188 original training set to 76,032 images.

Descriptor	Exp 1	Exp 2
Training Samples for VGG-19	19008	76032
VGG-19	62.4	68.3
RIC-LBP	-	48.1
JAMBP	-	50.1
JML	-	41.6
MP	-	44.6
VGG-19 + RIC-LBP	62.6	68.3
VGG-19 + RIC-LBP + JAMBP	63.9	68.4
VGG-19 + RIC-LBP + JAMBP + JML	64.6	68.3
VGG-19 + RIC-LBP + JAMBP + JML + MP	64.6	**68.5**

Table 5.11: Results of applying our framework on the newly augmented KTH-TIPS-2b using 1000 trees RF classifier. As we can see, the results in general are better than the corresponding Exp 1 results. We have obtained 68.3% using VGG-19 'fc7' features only. However, we could not improve a lot when we combined our local features with deep features.

The VGG-19 training parameters in this experiment are the same as the previous experiments with MinBatchSize of 40, learning rate of 0.001, and 10 Epochs. The results of end-to-end accuracy after training the network is 68.9%. This is higher than the figure obtained in experiment 1. One reason is because the network is now trained well to recognize the whole object in the texture image. However, when we extracted the 'fc7' features and used the RF-1000 classifier, we have only obtained 68.3%. We managed to improve that to only 68.5%. Hence, local features did not improve over the global deep features extracted from 'fc7' layer.

- **Experiment 3**:

  Experiment 2 demonstrated how powerful and important image augmentation for deep learning. Overall accuracy for both end-to-end learning and feature extraction and classification was improved due to the good training of VGG-19 architecture. In this experiment, we are trying to use additional augmentation techniques in addition to translation and rotation. Hence, we add 3 scales, 16 translations, 4 rotations, and 2 flips as illustrated in Figure 5.7. As a result we generate more images, specifically, we generate 456,192 (224×224) patches out of the 1188 training images. Hence, each image will contribute to 384 patches for the training stage. As we did with the previous two experiments, we used the RF classifier with 1000 trees after extracting the 'fc7' layer features. Results of this experiment are as listed in Table 5.12.

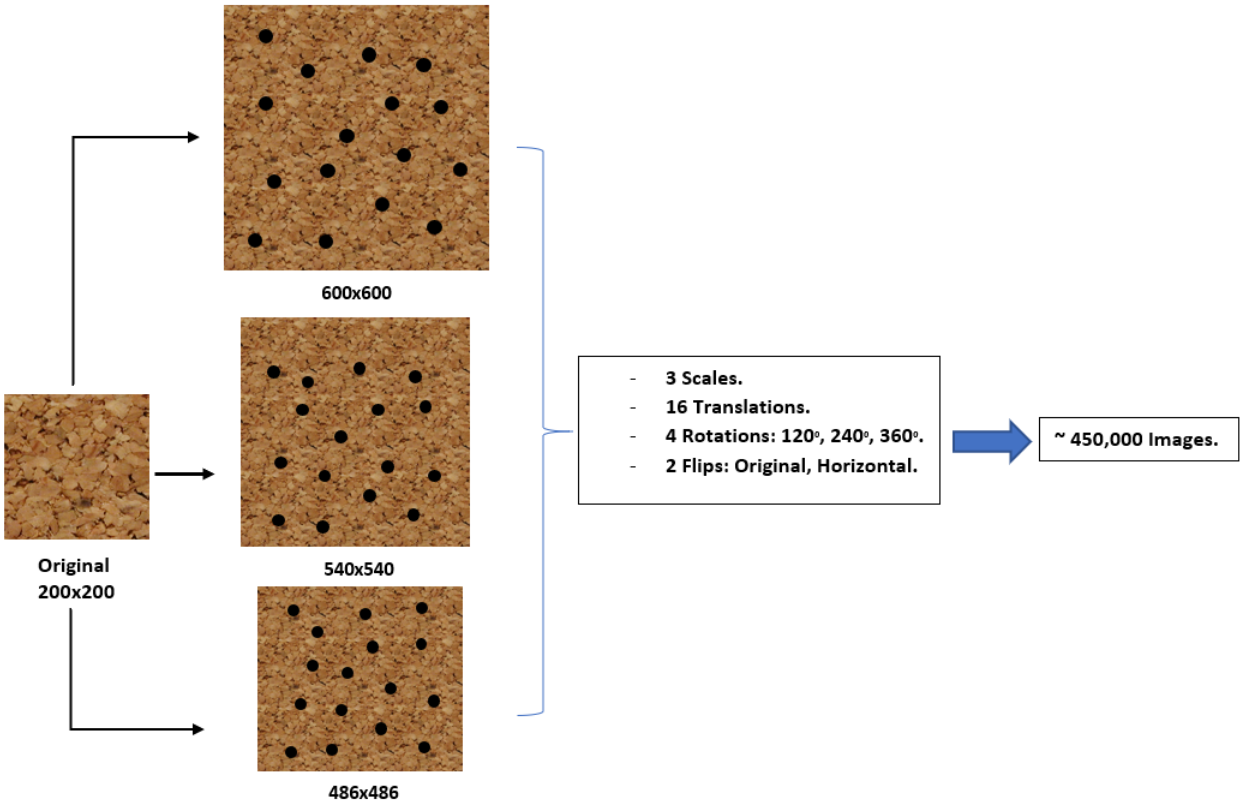


Figure 5.7: Thorough augmentation technique applied to the original KTH-TIPS-2b dataset. After synthesizing 600×600 large image and align the original image inside it, we start extracting 224×224 random patches (using the black dots as centers of these patches). Then, we perform the augmentation using translation, rotation, and rotation. As we can see 3 different scales are used to capture more texture spots of the original image. As a result, we extract 456,192 patches to be used in the training stage.

Descriptor	Exp 1	Exp 2	Exp 3
Training Samples for VGG-19	19 K	76 K	456 K
VGG-19	62.4	68.3	69.2
RIC-LBP	48.1	-	-
JAMBP	50.1	-	-
JML	41.6	-	-
MP	44.6	-	-
VGG-19 + RIC-LBP	62.6	68.3	69.2
VGG-19 + RIC-LBP + JAMBP	63.9	68.4	69.25
VGG-19 + RIC-LBP + JAMBP + JML	64.6	68.3	**69.68**
VGG-19 + RIC-LBP + JAMBP + JML + MP	64.6	68.5	69.2

Table 5.12: Results of applying our framework on the thoroughly augmented KTH-TIPS-2b dataset. RF classifier with 1000 trees was used for the classification. As we can see, the results are better than the previous two experiments. We have obtained 69.2% for 'fc7' layer features, in addition, we improved to 69.68% when we combined VGG-19 features with RIC-LBP, JAMBP, and JML features.

The training parameters for our model are still the same as the previous two experiments. The result of end-to-end accuracy after training the network is 68.9%. This is higher than the results obtained in both experiments 1 and 2. The reason behind that is the additional augmentation techniques that we used to sample more patches form the dataset. We can also notice that 'fc7' features classification performance with 1000 RFs was lower than end-to-end accuracy. Moreover, adding the local features to the deep features could not improve the accuracy to beat the end-to-end learning accuracy in this experiment.

After we obtained all these results using different augmentation techniques, we wanted to investigate the confusion matrix and see why we still obtain results with low accuracy. As we can see in Table 5.13, only three classes perform the worst with very low accuracy. These classes are: Cotton, Cracker, and Wool. This is because, the training and testing procedure used specifically for KTH-TIPS-2b dataset requires using only one quarter of the dataset for training and the rest for testing. Moreover, there are so many differences between images inside these samples in terms of shape,

color, and texture. This is the reason that makes it difficult for the classifier or the deep learning to recognize these different samples since we only provide fewer samples of data during the training stage.

	aluminium_foil	brown_bread	corduroy	cork	cotton	cracker	lettuce_leaf	linen	white_bread	wood	wool
aluminium_foil	**97.5**	0	0	0	0	0	1.3	0	1.2	0	0
brown_bread	0	**90**	0	8	0	1	0	0	2	0	0
corduroy	0	0	**92**	7	0	0	0	0	0	1	0
cork	0	0	0	**87**	0	10	0	0	3	0	0
cotton	0	0	22	0	**11**	0	0	44	1	2	20
cracker	0	42	4	2	0	**32**	0	4	15	0	0
lettuce_leaf	0	0	0	0	2	0	**98**	0	1	0	0
linen	0	0	2	0	15	0	0	**83**	0	1	0
white_bread	0	3	0	0	0	15	0	0	**82**	0	0
wood	0	0	0	0	3	0	0	0	2	**93**	2
wool	33	0	21	0	0	0	0	45	0	0	**1**

Table 5.13: Confusion Matrix: 11 Classes TIPS2b - MCA 69.68% using VGG-19 + RIC-LBP + JAMBP + JML features.

5.4.4 Comparison with state-of-the-art

Finally, a comparison with state-of-the-art methods is made in Table 5.14. On KTH-TIPS-2a dataset, Chen et al. performs 56.4% using WLD descriptor that is based on Webber's Law. Rahtu et al. achieves 67.7% using LPQ descriptor that is based on quantizing the information phase of the local Fourier transform. Recent work done by Khan et al. which is based upon combining different texture descriptors along with color and put then in a compact representation achieves an accuracy of 82.7%. On the other hand, our work of combining only four descriptors achieves 79.2% without using color information.

For the KTH-TIPS-2b dataset, both VZ-MR8 and VZ-Joint presented by Varma et al. perform 46.3 and 53.5 respectively. Binary Gabor Filter (BGP) descriptor presented by Zhang et al. based on convolving image with Gabor filters achieves 63.3%. Our work achieves 67.2 which is comparable to the work of Khan et al. using the five different descriptors and color that performs the best of 70.6%.

On the DTD dataset which is the most recent and challenging one. The best reported result using a local descriptor is the work of Nguyen et al. that performs

26.38% by applying LBP to a series of moment images. On the other hand, our work significantly outperforms the best local descriptor using SVM with additive chi-square kernel by achieving 43.5% which is 17% better than CSBP. However, Cimpoi et al. used a combination of Fisher Vector (FV) and DeCAF features and achieved 66.7%. Our result is still the best among the local descriptors and it is also worthy to mention that using different SVM kernels can improve the accuracy.

Method	TIPS-2a	TIPS-2b	DTD
LBP	-	52	14.51
WLD [121]	56.4	-	-
TFT [122]	-	66.3	-
LQP [123]	64.2	-	-
CSBP [124]	-	-	26.38
CLBP [48]	76.1	55.0	20.40
LVCBP [125]	61.7	53.6	-
VZ-MR8 [18]	-	46.3	-
VZ-Joint [21]	-	53.5	-
LTP [126]	60	-	-
LPQ [127]	67.7	54.4	-
BSIF [128]	70.0	54.3	-
BGP [129]	76.8	63.3	-
CLBP+WLD	78.1	63.7	-
CLBP+WLD+BGP	79.2	65.1	-
CLBP+WLD+BGP+LPQ	79.9	67.6	-
CLBP+WLD+BGP+LPQ+BSIF	80.8	68.9	-
Compact[CLBP+WLD+BGP+LPQ+BSIF]	82.2	69.0	-
Compact[CLBP+WLD+BGP+LPQ+BSIF]+Color	**82.7**	**70.6**	-
Our results	79.2	67.2	**43.5**

Table 5.14: Comparison of our approach with other state of the art methods. All these features are local features, in addition, some methods used a combination of five local descriptors besides color to obtain high accuracy for both KTH-TIPS2a, b datasets. In general, our results are comparable to the state-of-the-art local features.

5.5 Additional Experiments on Leaf Recognition and xVeiw Datasets

The main contribution in this thesis is to apply new texture descriptors to standard texture dataset and biomedical datasets. In addition to these applications, we started working on other applications including leaf recognition and classifying images from a standard and large dataset called xView. We have to emphasize that these experiments are still preliminary and we do not use our entire pipeline due to time limitations.

Experiments on Leaf Recognition Dataset

One of my esteemed committee member, Dr. David Larsen, suggested applying some classification techniques on a new dataset that he collected using his own professional camera from different areas on Missouri, USA. We managed to categorize 23 different species of leaves from different trees as shown in Figure 5.8. The problem is we only have very few number of images to work on. Some classes had two images and at most, some of them contain ten images. Due to time limitations, I applied only deep learning methods, specifically VGG-19 end-to-end learning. I used augmentation techniques by augmenting the dataset using rotation. Moreover, I split the dataset into two splits because we have fewer number of images. The results that we obtained using VGG-19 end-to-end learning are as listed in Table 5.15.

Method	Epochs	Efficiency in Mins	Accuracy
VGG-19	10	37	37.8
VGG-19	30	112	**40.2**
VGG-19	50	-	Out of Cache Memory

Table 5.15: Results of using only VGG-19 architecture on the leaf recognition dataset. The highest accuracy we obtained was only 40.2% using image augmentation. This is because we are only provided with fewer number of images per class.

Figure 5.8: 23 classes of leaves captured from different regions of Missouri, USA.

Experiments on xView Dataset

In 2018, a large-scale object detection dataset was introduce by Lam et al. [132]. It contains 60 classes, and was collected from WorldView satellites at 0.3m ground sample distance in order to provide higher resolution imagery. For our experiments, we selected only 10 classes as in Figure 5.9. In the experiments, we extracted 7804 bounding boxes from these 10 classes and used subset of our features (RIC-LBP + JML). Random Forests classifier was used with 1000 trees in the classification phase along with five-fold cross-validation. The accuracy we obtained using these two set of features was 49.7% for the 10 classes.

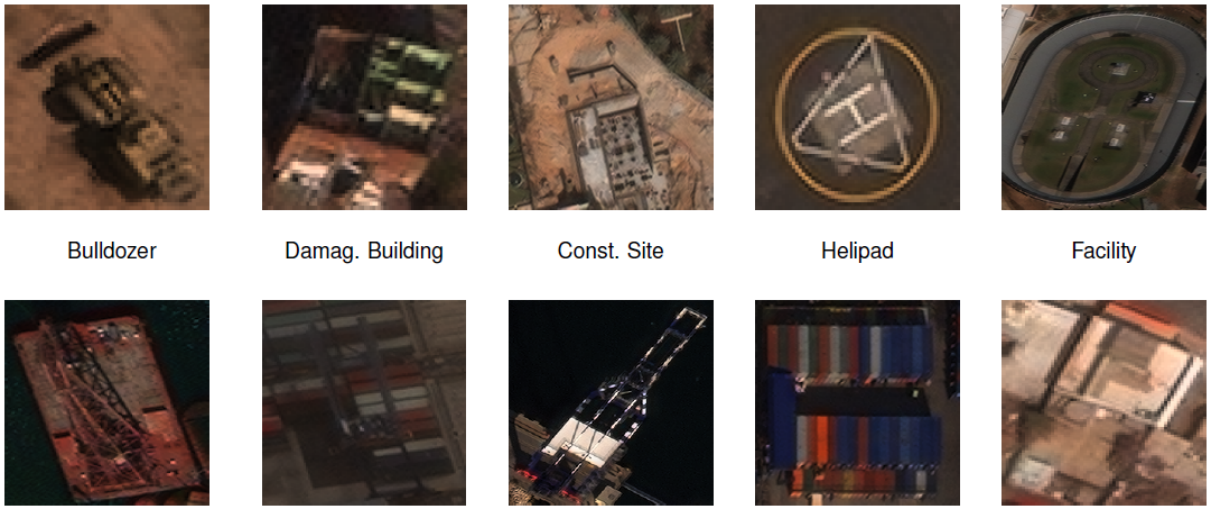


Figure 5.9: 10 xView classes used in our experiments. We have selected these classes to see if we can capture useful texture features and classify them successfully.

Chapter 6

Summary and concluding remarks

This thesis has presented a new approach for texture classification using a combination of deep and local descriptors. The deep features used are extracted from the Fully-Connected layer ('fc7') of VGG-19 network. The size of these features is 4096 bins and a Random Forests (RF) classifier was applied to these features in the classification stage. For the local descriptors, we have proposed a set of four operators. The descriptors used are derived from standard texture descriptors called Local Binary Pattern (LBP) and Motif Co-occurrence Matrix (MCM). Two powerful LBP variations are utilized with two new motif-related descriptors and all four descriptors were used to classify HEp-2 cell-specimen images and the standard texture datasets. LBP variations has been used in the past two decades in different applications including biomedical analysis and showed how powerful LBP-based descriptors in terms of achieving high classification accuracy. However, MCM was used only for image indexing and retrieval and has not been used for texture classification purposes. In this work, we developed two descriptors called Joint Motif Labels (JML) and Motif Patterns (MP) which are based on MCM descriptors and used them for HEp-2 analysis and texture classification.

In the experiments, we have applied our framework to an important application

in the field of pattern analysis and computer vision which is HEp-2 cell and specimen classification. We have also considered applying our framework on challenging texture databases where each database consists of thousands of images with varying shapes, scale, color, and texture. We showed that our framework work very well on HEp-2 cell and specimen images by outperforming all the state-of-the-art methods using 1000 trees RF classifier. We have also shown that the proposed local descriptors along with the existent methods utilized in our framework achieve high performance in comparison with other features. The performance of our framework on standard texture databases did not achieve high accuracy in comparison to the state-of-the-art but, it is promising.

For the future work, we intend to apply our approach on other applications like leaf recognition and stromal and eipthelial classification tasks. We will also consider different augmentation methods in order to improve the classification results with deep learning. In addition, researches in the literature have shown that other features from different deep layers can also be used. For example, we can extract convolutional features and quantize them and apply the RF classifier especially for the texture classification task.

Bibliography

[1] N Jhanwar, Subhasis Chaudhuri, Guna Seetharaman, and Bertrand Zavidovique. Content based image retrieval using motif cooccurrence matrix. *Image and Vision Computing*, 22(14):1211–1220, 2004.

[2] Ryusuke Nosaka and Kazuhiro Fukui. Hep-2 cell classification using rotation invariant co-occurrence among local binary patterns. *Pattern Recognition*, 47(7):2428–2436, 2014.

[3] Adel Hafiane, Kannappan Palaniappan, and Guna Seetharaman. Joint adaptive median binary patterns for texture classification. *Pattern Recognition*, 48(8):2609–2620, 2015.

[4] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[5] Robert M Haralick, Karthikeyan Shanmugam, et al. Textural features for image classification. *IEEE Transactions on systems, man, and cybernetics*, (6):610–621, 1973.

[6] Hideyuki Tamura, Shunji Mori, and Takashi Yamawaki. Textural features corresponding to visual perception. *IEEE Transactions on Systems, man, and cybernetics*, 8(6):460–473, 1978.

[7] A. Ravishankar Rao and Gerald L. Lohse. Identifying high level features of texture perception. *CVGIP: Graphical Models and Image Processing*, 55(3):218–233, 1993.

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

[9] Jianguo Zhang, Marcin Marszałek, Svetlana Lazebnik, and Cordelia Schmid. Local features and kernels for classification of texture and object categories: A comprehensive study. *International journal of computer vision*, 73(2):213–238, 2007.

[10] Matti Pietikäinen, Abdenour Hadid, Guoying Zhao, and Timo Ahonen. *Computer vision using local binary patterns*, volume 40. Springer Science & Business Media, 2011.

[11] Li Liu, Paul Fieguth, Xiaogang Wang, Matti Pietikäinen, and Dewen Hu. Evaluation of lbp and deep texture descriptors with a new robustness benchmark. In *European Conference on Computer Vision*, pages 69–86. Springer, 2016.

[12] Timo Ojala, Matti Pietikäinen, and David Harwood. A comparative study of texture measures with classification based on featured distributions. *Pattern recognition*, 29(1):51–59, 1996.

[13] David Harwood, Timo Ojala, Matti Pietikäinen, Shalom Kelman, and Larry Davis. Texture classification by center-symmetric auto-correlation, using kullback discrimination of distributions. *Pattern Recognition Letters*, 16(1):1–10, 1995.

[14] Kenneth Abend, Tl Harley, and L Kanal. Classification of binary random patterns. *IEEE Transactions on Information Theory*, 11(4):538–544, 1965.

[15] Theodore Bially. Space-filling curves: Their generation and their application to bandwidth reduction. *IEEE Transactions on Information Theory*, 15(6):658–664, 1969.

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

[17] Jorge Sánchez, Florent Perronnin, Thomas Mensink, and Jakob Verbeek. Image classification with the fisher vector: Theory and practice. *International journal of computer vision*, 105(3):222–245, 2013.

[18] Manik Varma and Andrew Zisserman. A statistical approach to texture classification from single images. *International journal of computer vision*, 62(1-2):61–81, 2005.

[19] Bangalore S Manjunath and Wei-Ying Ma. Texture features for browsing and retrieval of image data. *IEEE Transactions on pattern analysis and machine intelligence*, 18(8):837–842, 1996.

[20] Thomas Leung and Jitendra Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *International journal of computer vision*, 43(1):29–44, 2001.

[21] Manik Varma and Andrew Zisserman. A statistical approach to material classification using image patch exemplars. *IEEE transactions on pattern analysis and machine intelligence*, 31(11):2032–2047, 2009.

[22] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[23] Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Return of the devil in the details: Delving deep into convolutional nets. *arXiv preprint arXiv:1405.3531*, 2014.

[24] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, and Andrea Vedaldi. Deep filter banks for texture recognition, description, and segmentation. *International Journal of Computer Vision*, 118(1):65–94, 2016.

[25] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[26] Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1717–1724, 2014.

[27] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 806–813, 2014.

[28] Dmitrii Chemodanov, Flavio Esposito, Andrei Sukhov, Prasad Calyam, Huy Trinh, and Zakariya Oraibi. Agra: Ai-augmented geographic routing approach for iot-based incident-supporting applications. *Future Generation Computer Systems*, 92:1051–1065, 2019.

[29] Rasha Gargees, Brittany Morago, Rengarajan Pelapur, Dmitrii Chemodanov, Prasad Calyam, Zakariya Oraibi, Ye Duan, Guna Seetharaman, and Kannappan

Palaniappan. Incident-supporting visual cloud computing utilizing software-defined networking. *IEEE Transactions on Circuits and Systems for Video Technology*, 27(1):182–197, 2016.

[30] Timo Ojala, Matti Pietikainen, and David Harwood. Performance evaluation of texture measures with classification based on kullback discrimination of distributions. In *Pattern Recognition, 1994. Vol. 1-Conference A: Computer Vision & Image Processing., Proceedings of the 12th IAPR International Conference on*, volume 1, pages 582–585. IEEE, 1994.

[31] Timo Ojala, Matti Pietikainen, and Topi Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on pattern analysis and machine intelligence*, 24(7):971–987, 2002.

[32] Larry S Davis, Steven A Johns, and JK Aggarwal. Texture analysis using generalized co-occurrence matrices. *IEEE Transactions on pattern analysis and machine intelligence*, (3):251–259, 1979.

[33] Dmitry Chetverikov. Experiments in the rotation-invariant texture discrimination using anisotropy features. In *Proceedings-International Conference on Pattern Recognition*. IEEE, 1982.

[34] Rangasami L Kashyap and Alireza Khotanzad. A model-based method for rotation invariant texture classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (4):472–481, 1986.

[35] Jianchang Mao and Anil K Jain. Texture classification and segmentation using multiresolution simultaneous autoregressive models. *Pattern recognition*, 25(2):173–188, 1992.

[36] Wen-Rong Wu and Shieh-Chung Wei. Rotation and gray-scale transform-invariant texture classification using spiral resampling, subband decomposi-

tion, and hidden markov model. *IEEE Transactions on Image Processing*, 5(10):1423–1434, 1996.

[37] Fernand S. Cohen, Zhigang Fan, and Maqbool A Patel. Classification of rotated and scaled textured images using gaussian markov random field models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(2):192–202, 1991.

[38] Stephanie R Fountain and TN Tan. Efficient rotation invariant texture features for content-based image retrieval. *Pattern Recognition*, 31(11):1725–1732, 1998.

[39] 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*, volume 2, pages 162–167. IEEE, 1994.

[40] George M Haley and BS Manjunath. Rotation-invariant texture classification using a complete space-frequency model. *IEEE transactions on Image Processing*, 8(2):255–269, 1999.

[41] W-K Lam and C-K Li. Rotated texture classification by improved iterative morphological decomposition. *IEE Proceedings-Vision, Image and Signal Processing*, 144(3):171–179, 1997.

[42] Michael M Leung and Allen M Peterson. Scale and rotation invariant texture classification. In *Signals, Systems and Computers, 1992. 1992 Conference Record of The Twenty-Sixth Asilomar Conference on*, pages 461–465. IEEE, 1992.

[43] MOSHE Porat and Yehoshua Y Zeevi. Localized texture processing in vision: Analysis and synthesis in the gaborian space. *IEEE Transactions on Biomedical Engineering*, 36(1):115–129, 1989.

[44] Olivier Alata, Claude Cariou, Clarisse Ramananjarasoa, and Mohamed Najim. Classification of rotated and scaled textures using hmhv spectrum estimation and the fourier-mellin transform. In *Image Processing, 1998. ICIP 98. Proceedings. 1998 International Conference on*, volume 1, pages 53–56. IEEE, 1998.

[45] Vidya Manian and Ramon Vasquez. Scaled and rotated texture classification using a class of basis functions. *Pattern Recognition*, 31(12):1937–1948, 1998.

[46] Jane You and Harvey A Cohen. Classification and segmentation of rotated and scaled textured images using texture "tuned" masks. *Pattern Recognition*, 26(2):245–258, 1993.

[47] Lizhi Wang and Glenn Healey. Using zernike moments for the illumination and geometry invariant classification of multispectral texture. *IEEE Transactions on Image Processing*, 7(2):196–203, 1998.

[48] Zhenhua Guo, Lei Zhang, and David Zhang. A completed modeling of local binary pattern operator for texture classification. *IEEE Transactions on Image Processing*, 19(6):1657–1663, 2010.

[49] Yang Zhao, De-Shuang Huang, and Wei Jia. Completed local binary count for rotation invariant texture classification. *IEEE transactions on image processing*, 21(10):4492–4497, 2012.

[50] Yimo Guo, Guoying Zhao, and Matti PietikäInen. Discriminative features for texture description. *Pattern Recognition*, 45(10):3834–3843, 2012.

[51] Timo Ahonen, Jiří Matas, Chu He, and Matti Pietikäinen. Rotation invariant image description with local binary pattern histogram fourier features. In *Scandinavian Conference on Image Analysis*, pages 61–70. Springer, 2009.

[52] Zhenhua Guo, Lei Zhang, and David Zhang. Rotation invariant texture classification using lbp variance (lbpv) with global matching. *Pattern recognition*, 43(3):706–719, 2010.

[53] Xiaoyang Tan and Bill Triggs. Enhanced local texture feature sets for face recognition under difficult lighting conditions. *IEEE transactions on image processing*, 19(6):1635–1650, 2010.

[54] Hui Zhou, Runsheng Wang, and Cheng Wang. A novel extended local-binary-pattern operator for texture analysis. *Information Sciences*, 178(22):4314–4325, 2008.

[55] Xianbiao Qi, Rong Xiao, Chun-Guang Li, Yu Qiao, Jun Guo, and Xiaoou Tang. Pairwise rotation invariant co-occurrence local binary pattern. *IEEE transactions on pattern analysis and machine intelligence*, 36(11):2199–2213, 2014.

[56] Xiaosheng Wu and Junding Sun. Joint-scale lbp: a new feature descriptor for texture classification. *The Visual Computer*, 33(3):317–329, 2017.

[57] Li Liu, Yunli Long, Paul W Fieguth, Songyang Lao, and Guoying Zhao. Brint: binary rotation invariant and noise tolerant texture classification. *IEEE transactions on Image Processing*, 23(7):3071–3084, 2014.

[58] Rakesh Mehta and Karen Egiazarian. Dominant rotated local binary patterns (drlbp) for texture classification. *Pattern Recognition Letters*, 71:16–22, 2016.

[59] Li Liu, Lingjun Zhao, Yunli Long, Gangyao Kuang, and Paul Fieguth. Extended local binary patterns for texture classification. *Image and Vision Computing*, 30(2):86–99, 2012.

[60] Shu Liao, Max WK Law, and Albert CS Chung. Dominant local binary patterns for texture classification. *IEEE transactions on image processing*, 18(5):1107–1118, 2009.

[61] Arthur R Butz. Space filling curves and mathematical programming. Technical report, NORTHWESTERN UNIV EVANSTON ILL INFORMATION-PROCESSING AND CONTROL SYSTEMS LAB, 1967.

[62] Guna Seetharaman, Bertrand Zavidovique, and Sashidar Shivayogimath. Z-trees: adaptive pyramid-algorithms for image segmentation. In *Image Processing, 1998. ICIP 98. Proceedings. 1998 International Conference on*, pages 294–298. IEEE, 1998.

[63] M Subrahmanyam, QM Jonathan Wu, RP Maheshwari, and R Balasubramanian. Modified color motif co-occurrence matrix for image indexing and retrieval. *Computers & Electrical Engineering*, 39(3):762–774, 2013.

[64] Chuen-Horng Lin and Wei-Chih Lin. Image retrieval system based on adaptive color histogram and texture features. *The Computer Journal*, 54(7):1136–1147, 2011.

[65] Chuen-Horng Lin, Rong-Tai Chen, and Yung-Kuan Chan. A smart content-based image retrieval system based on color and texture feature. *Image and Vision Computing*, 27(6):658–665, 2009.

[66] Santosh Kumar Vipparthi and SK Nagar. Expert image retrieval system using directional local motif xor patterns. *Expert Systems with Applications*, 41(17):8016–8026, 2014.

[67] L Koteswara Rao, D Venkata Rao, and L Pratap Reddy. Color based multi directional local motif xor patterns for image retrieval. In *Applied and Theoret-*

ical Computing and Communication Technology (iCATccT), 2015 International Conference on, pages 852–856. IEEE, 2015.

[68] Santosh Kumar Vipparthi and Shyam Krishna Nagar. Multi-joint histogram based modelling for image indexing and retrieval. *Computers & Electrical Engineering*, 40(8):163–173, 2014.

[69] Santosh Kumar Vipparthi, Subrahmanyam Murala, and Shyam Krishna Nagar. Dual directional multi-motif xor patterns: A new feature descriptor for image indexing and retrieval. *Optik-International Journal for Light and Electron Optics*, 126(15-16):1467–1473, 2015.

[70] David G Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee, 1999.

[71] John A Hartigan and Manchek A Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108, 1979.

[72] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[73] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[74] Stephen I Gallant. Perceptron-based learning algorithms. *IEEE Transactions on neural networks*, 1(2):179–191, 1990.

[75] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1422–1430, 2015.

[76] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal processing magazine*, 29(6):82–97, 2012.

[77] Kevin Gurney. *An introduction to neural networks*. CRC press, 2014.

[78] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[79] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[80] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[81] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[82] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, volume 1, page 3, 2017.

[83] Mircea Cimpoi, Subhransu Maji, and Andrea Vedaldi. Deep filter banks for texture recognition and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3828–3836, 2015.

[84] Paolo Napoletano. Hand-crafted vs learned descriptors for color texture classification. In *International Workshop on Computational Color Imaging*, pages 259–271. Springer, 2017.

[85] Florent Perronnin, Jorge Sánchez, and Thomas Mensink. Improving the fisher kernel for large-scale image classification. In *European conference on computer vision*, pages 143–156. Springer, 2010.

[86] Herve Jegou, Florent Perronnin, Matthijs Douze, Jorge Sánchez, Patrick Perez, and Cordelia Schmid. Aggregating local image descriptors into compact codes. *IEEE transactions on pattern analysis and machine intelligence*, 34(9):1704–1716, 2012.

[87] Jinjun Wang, Jianchao Yang, Kai Yu, Fengjun Lv, Thomas Huang, and Yihong Gong. Locality-constrained linear coding for image classification. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3360–3367. IEEE, 2010.

[88] Yang Song, Fan Zhang, Qing Li, Heng Huang, Lauren J O'Donnell, and Weidong Cai. Locally-transferred fisher vectors for texture classification. In *ICCV*, pages 4922–4930, 2017.

[89] Leon Gatys, Alexander S Ecker, and Matthias Bethge. Texture synthesis using convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 262–270, 2015.

[90] Tsung-Yu Lin and Subhransu Maji. Visualizing and understanding deep texture representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2791–2799, 2016.

[91] Vincent Andrearczyk and Paul F Whelan. Using filter banks in convolutional neural networks for texture classification. *Pattern Recognition Letters*, 84:63–69, 2016.

[92] Yang Gao, Oscar Beijbom, Ning Zhang, and Trevor Darrell. Compact bilinear pooling. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 317–326, 2016.

[93] Tsung-Yu Lin, Aruni RoyChowdhury, and Subhransu Maji. Bilinear convolutional neural networks for fine-grained visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 40(6):1309–1322, 2018.

[94] Joan Bruna and Stéphane Mallat. Invariant scattering convolution networks. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1872–1886, 2013.

[95] Laurent Sifre and Stéphane Mallat. *Rigid-motion scattering for image classification*. PhD thesis, Citeseer, 2014.

[96] Laurent Sifre and Stéphane Mallat. Combined scattering for rotation invariant texture analysis. In *ESANN*, volume 44, pages 68–81, 2012.

[97] Pasquale Foggia, Gennaro Percannella, Paolo Soda, and Mario Vento. Benchmarking hep-2 cells classification methods. *IEEE transactions on medical imaging*, 32(10):1878–1889, 2013.

[98] National Committee for Clinical Laboratory Standards and Robert M Nakamura. *Quality Assurance for the Indirect Immunofluorescence Test for Autoantibodies to Nuclear Antigen (IF-ANA): Approved Guideline*. NCCLS, 1996.

[99] Amelia Rigon, Paolo Soda, Danila Zennaro, Giulio Iannello, and Antonella Afeltra. Indirect immunofluorescence in autoimmune diseases: assessment of digital images for diagnostic purpose. *Cytometry Part B: Clinical Cytometry*, 72(6):472–477, 2007.

[100] Santa Di Cataldo, Andrea Bottino, Elisa Ficarra, and Enrico Macii. Applying textural features to the classification of hep-2 cell patterns in iif images. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 3349–3352. IEEE, 2012.

[101] Ilker Ersoy, Filiz Bunyak, Jing Peng, and Kannappan Palaniappan. Hep-2 cell classification in iif images using shareboost. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 3362–3365. IEEE, 2012.

[102] Kuan Li, Jianping Yin, Zhi Lu, Xiangfei Kong, Rui Zhang, and Wenyin Liu. Multiclass boosting svm using different texture features in hep-2 cell staining pattern classification. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 170–173. IEEE, 2012.

[103] Anders Boesen Lindbo Larsen, Jacob Schack Vestergaard, and Rasmus Larsen. Hep-2 cell classification using shape index histograms with donut-shaped spatial pooling. *IEEE transactions on medical imaging*, 33(7):1573–1580, 2014.

[104] Ryusuke Nosaka, Yasuhiro Ohkawa, and Kazuhiro Fukui. Feature extraction based on co-occurrence of adjacent local binary patterns. In *Pacific-Rim Symposium on Image and Video Technology*, pages 82–91. Springer, 2011.

[105] Diego Gragnaniello, Carlo Sansone, and Luisa Verdoliva. Biologically-inspired dense local descriptor for indirect immunofluorescence image classification. In *Pattern Recognition Techniques for Indirect Immunofluorescence Images (I3A), 2014 1st Workshop on*, pages 1–5. IEEE, 2014.

[106] VB Surya Prasath, Yasmin M Kassim, Zakariya A Oraibi, Jean-Baptiste Guiriec, Adel Hafiane, Guna Seetharaman, and Kannappan Palaniappan. Hep-2 cell classification and segmentation using motif texture patterns and spatial features with random forests. In *Pattern Recognition (ICPR), 2016 23rd International Conference on*, pages 90–95. IEEE, 2016.

[107] Shahab Ensafi, Shijian Lu, Ashraf A Kassim, and Chew Lim Tan. A bag of words based approach for classification of hep-2 cell images. In *Pattern Recognition Techniques for Indirect Immunofluorescence Images (I3A), 2014 1st Workshop on*, pages 29–32. IEEE, 2014.

[108] Ilias Theodorakopoulos, Dimitris Kastaniotis, George Economou, and Spiros Fotopoulos. Hep-2 cells classification via fusion of morphological and textural features. In *Bioinformatics & Bioengineering (BIBE), 2012 IEEE 12th International Conference on*, pages 689–694. IEEE, 2012.

[109] Xi Jia, Linlin Shen, Xiande Zhou, and Shiqi Yu. Deep convolutional neural network based hep-2 cell classification. In *Pattern Recognition (ICPR), 2016 23rd International Conference on*, pages 77–80. IEEE, 2016.

[110] Yuexiang Li, Linlin Shen, Xiande Zhou, and Shiqi Yu. Hep-2 specimen classification with fully convolutional network. In *Pattern Recognition (ICPR), 2016 23rd International Conference on*, pages 96–100. IEEE, 2016.

[111] Siyamalan Manivannan, Wenqi Li, Shazia Akbar, Ruixuan Wang, Jianguo Zhang, and Stephen J McKenna. Hep-2 cell classification using multi-resolution

local patterns and ensemble svms. In *Pattern Recognition Techniques for Indirect Immunofluorescence Images (I3A), 2014 1st Workshop on*, pages 37–40. Ieee, 2014.

[112] Iasonas Kokkinos, Michael Bronstein, and Alan Yuille. *Dense scale invariant descriptors for images and surfaces*. PhD thesis, INRIA, 2012.

[113] Zakariya A Oraibi, Morgane Irio, Adel Hafiane, and Kannappan Palaniappan. Texture classification using multiple local descriptors. In *2017 IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*, pages 1–7. IEEE, 2017.

[114] Zakariya A Oraibi, Hayder Yousif, Adel Hafiane, Guna Seetharaman, and Kannappan Palaniappan. Learning local and deep features for efficient cell image classification using random forests. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 2446–2450. IEEE, 2018.

[115] P Mallikarjuna, M Fritz, A Tavakoli Targhi, E Hayman, B Caputo, and JO Eklundh. The kth-tips and kth-tips2 databases, 2006.

[116] Nalini Bhushan, A Ravishankar Rao, and Gerald L Lohse. The texture lexicon: Understanding the categorization of visual texture terms and their relationship to texture images. *Cognitive Science*, 21(2):219–246, 1997.

[117] Naomi S Altman. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185, 1992.

[118] Tin Kam Ho. Random decision forests. In *Document analysis and recognition, 1995., proceedings of the third international conference on*, volume 1, pages 278–282. IEEE, 1995.

[119] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[120] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[121] Jie Chen, Shiguang Shan, Chu He, Guoying Zhao, Matti Pietikainen, Xilin Chen, and Wen Gao. Wld: A robust local image descriptor. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1705–1720, 2010.

[122] Radu Timofte and Luc J Van Gool. A training-free classification framework for textures, writers, and materials. In *BMVC*, volume 13, page 14, 2012.

[123] Sibt ul Hussain and Bill Triggs. Visual recognition using local quantized patterns. In *Computer Vision–ECCV 2012*, pages 716–729. Springer, 2012.

[124] Thanh Phuong Nguyen, Ngoc-Son Vu, and Antoine Manzanera. Statistical binary patterns for rotational invariant texture classification. *Neurocomputing*, 173:1565–1577, 2016.

[125] Seung Ho Lee, Jae Young Choi, Yong Man Ro, and Konstantinos N Plataniotis. Local color vector binary patterns from multichannel face images for face recognition. *IEEE Transactions on Image Processing*, 21(4):2347–2353, 2012.

[126] Xiaoyang Tan and Bill Triggs. Enhanced local texture feature sets for face recognition under difficult lighting conditions. *Analysis and modeling of faces and gestures*, pages 168–182, 2007.

[127] Esa Rahtu, Janne Heikkilä, Ville Ojansivu, and Timo Ahonen. Local phase quantization for blur-insensitive image analysis. *Image and Vision Computing*, 30(8):501–512, 2012.

[128] Juho Kannala and Esa Rahtu. Bsif: Binarized statistical image features. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 1363–1366. IEEE, 2012.

[129] Lin Zhang, Zhiqiang Zhou, and Hongyu Li. Binary gabor pattern: An efficient and robust descriptor for texture classification. In *Image Processing (ICIP), 2012 19th IEEE International Conference on*, pages 81–84. Ieee, 2012.

[130] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 675–678. ACM, 2014.

[131] Siyamalan Manivannan, Wenqi Li, Shazia Akbar, Ruixuan Wang, Jianguo Zhang, and Stephen J McKenna. An automated pattern recognition system for classifying indirect immunofluorescence images of hep-2 cells and specimens. *Pattern Recognition*, 51:12–26, 2016.

[132] Darius Lam, Richard Kuzma, Kevin McGee, Samuel Dooley, Michael Laielli, Matthew Klaric, Yaroslav Bulatov, and Brendan McCord. xview: Objects in context in overhead imagery. *arXiv preprint arXiv:1802.07856*, 2018.

VITA

I was born in Basrah, Iraq in 1985. I graduated from the college of science/ university of Basrah with a bachelor and master degrees in computer science in 2007 and 2010 respectively.

In late 2011, I was appointed as assistant teacher in college of education/ computer science department at the same university. Before that, I was accepted as a candidate to pursue a PhD degree in the computer science field in 2010. Later on, after finishing all the paper work, I joined the program in the United States in 2014.

Upon arrival to Columbia, Missouri in January/2014 to pursue my PhD scholarship, I started learning English at the Intensive English Program (IEP) at MU. After fulfilling the requirements to be admitted as a PhD candidate at the computer science department at MU, I chose my adviser professor Kannappan Palaniappan.

We began planning for the research by taking the required and related classes and was motivated by his ideas in texture classification to device a new approach and apply it in biomedical image analysis. We were capable of creating a new pipeline to classify human cells successfully, moreover, we took part in other projects that won grants from the federal government of USA.

After defending my dissertation and complete the graduation requirements, I will return back to my country and resume working as a lecturer in my university (University of Basrah in Basrah Province). In addition, I will continue doing research especially in biomedical image analysis and segmentation.