STRUCTURAL TEXTURE IMAGE RECOGNITION BASED ON TEXTURE FEATURES AND DEEP LEARNING ALGORITHMS

Tran Thanh Hung[1,*], Vu Thi Duong[1]

DOI: https://doi.org/10.57001/huih5804.2025.425

ABSTRACT

Texture image recognition is a critical task with numerous practical applications, yet it often faces challenges due to environmental conditions such as lighting, viewing angle, and scale. This study proposes an effective texture recognition method utilizing a texture descriptor based on local ZigZag patterns, combined with directional edge information extracted from Kirsch masks in six orientations. Hyperparameter tuning techniques are applied within the deep learning framework to enhance classification performance. Experimental results on the KTH-TIPS dataset demonstrate that the proposed method achieves an accuracy of 99.97%. Compared to traditional methods and various other deep learning models, this approach exhibits superior accuracy. Furthermore, it proves robust against lighting variations and noise, making it suitable for high-precision texture recognition applications.

***Keywords*:** *Image configuration, local ZigZag sample set, hyperparameter tuning, classification configuration, KTH-TIPS.*

[1]School of Information and Communications Technology, Hanoi University of Industry, Vietnam

[*]Email: hungtt@fit-haui.edu.vn

Received: 10/8/2025
Revised: 05/10/2025
Accepted: 28/11/2025

1. INTRODUCTION

Structure is defined as an arrangement and organization of elements within an object or a system, or such an organized system or object itself. Structures can be man-made, such as factories, buildings, tables, and chairs, or natural, like the shapes of minerals and other objects. In the context of images, "image texture" refers to the visual manifestation of how pixels are arranged in a picture to form a specific shape.

Image texture is often repeating patterns that characterize the type of material the object possesses. Texture is one of the important characteristics of an image that helps algorithms to segment images into regions of interest and classify those regions. In some images, characteristics of the region can be defined to obtain accurate analyses. Although we can perceive some characteristics of the image such as image smoothness, image depth, image brightness, we do not have a complete definition of texture. Many researchers in the field have proposed different definitions for texture. In paper [1] defines image texture as a set of descriptors of the brightness variations from one pixel to another in a small neighboring region across the entire image. Meanwhile, the author of paper [2] defines texture as a spatial arrangement of grayscale values within a region of an image. The most prominent applications of texture images can be listed as medical image analysis, remote sensing image analysis, and product defect recognition in industrial production lines. In texture image processing, there are three most important fields: texture image classification, texture image segmentation and texture image synthesis. Texture image classification relates to building algorithms to classify input images into corresponding labels of the material textures of the objects within the image. Texture image segmentation relates to building algorithms to divide an input image into smaller regions corresponding to the respective texture labels. Meanwhile, texture image synthesis relates to creating texture images based on given statistical models. Approaches to the problem of texture analysis in images are highly diverse, mainly differing in how they perform the texture feature extraction. The four main types of feature extraction stages can be listed as: Statistical-based methods; Structural-based methods; Model-based methods; and Transform-based methods.

Statistical-based methods for analyzing image texture describe the characteristics of texture regions in an image through the distribution of their grayscale histograms.

One of the most well-known statistical methods in texture image analysis is feature extraction from the Gray Level Co-occurrence Matrix (GLCM) [3]. This GLCM method relies on using second-order statistics of the grayscale histogram in the image. Besides using traditional statistical texture analysis, multivariate statistical methods have also been proposed for this texture feature extraction problem. Viewing an image as a matrix, the SVD (Singular Value Decomposition) spectrum is the composite vector of the texture in the image, represented by its singular values. The SVD spectrum is used as a feature vector for the texture classification problem.

Structural-based methods for analyzing image texture describe a texture as a combination of well-defined textural primitives, such as parallel lines spaced apart. The characteristics and positions of these textural elements combine to form the image textures. Many approaches to texture analysis based on structural elements have been proposed, from using different shapes of textural primitives to treating real-world textures as distorted versions of ideal textures. However, these methods are often limited in practice because they can only describe textures and lack the ability to recognize or analyze complex textures [4].

Model-based methods for analysis create empirical models of each pixel in an image based on the weighted average of the grayscale values of pixels in its neighborhoods. The estimated parameters of the image model are used as texture feature descriptors. Other texture feature descriptors used in some studies include Autoregressive (AR) [5] and Markov Random Field (MRF) [6].

Finally, transform-based methods for texture analysis convert an image into other forms using the spatial frequency characteristics of the grayscale variations of pixels. This approach heavily relies on the quality and perspective of the transformation method used to extract texture features from the image. The authors in [7] describe the use of the spectrum from a 2D Fourier Transform (2D FFT) to extract texture features. Other transformation methods also used for feature extraction include Gabor transforms and Wavelet transforms. Features derived from Gabor filters are widely used in image texture analysis for the image segmentation problem. Wavelet transforms are also utilized to extract texture features for addressing the image segmentation problem.

Based on these studies, it can be seen that analyzing the texture of materials in images is a promising research direction, with many practical applications that can be applied to various problems. Figure 1 shows some examples of material texture images from the KTH-TIPS texture dataset [8].

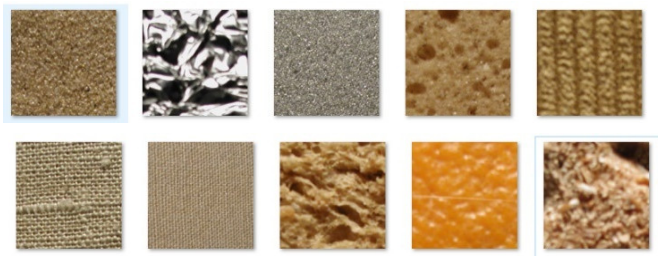

Figure 1. Illustrative images from the KTH-TIPS texture dataset

In this study, we use a ZigZag scanning-based feature extraction method on a texture image dataset to generate an image texture descriptor. Next, a hyperparameter tuning method in a deep learning model is employed to improve the accuracy of the texture image recognition problem.

2. TEXTURE IMAGE RECOGNITION

In the texture image recognition problem, the ultimate goal is to assign an unknown image sample to one of the predefined texture classes. As mentioned in the previous section, texture image recognition is one of the four main research domains in the field of texture image analysis. The process of recognizing texture in images involves two main phases: the training phase and the recognition phase. In the training phase, the goal is to build a model for the textural content of each texture class in the training data, which includes images with known labels. The textural content of the images in the training set is collected using pre-selected texture analysis methods. These texture image analysis methods will return a set of feature vectors that represent each different texture type. These features can be integers, grayscale distribution charts, or empirical distributions of the images. These features are used to represent the characteristics of the texture in the image, such as spatial structure, contrast, roughness, and texture direction. In the recognition phase, the textural content of the unknown samples is first described using the same analysis methods. Then, these features from the unknown samples are compared with the samples in the training set using classification algorithms, and the unknown samples are assigned to the class with the highest degree of similarity. As an alternative option when performing the texture image recognition task, if a

sample's highest similarity with a class is not good enough according to some predefined criteria, these samples may be excluded from the initial classification goal. The process of building a texture image classification model is shown in Figure 2.

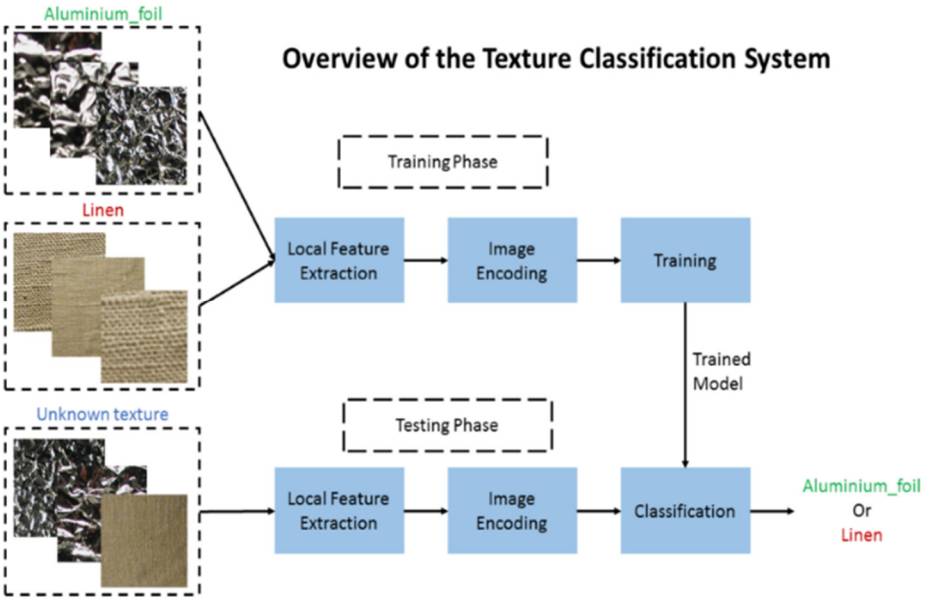

Figure 2. Overview of the texture classification system

When selecting a texture analysis algorithm, several aspects should be considered for the best results:

- Invariance to illumination: This is the algorithm's sensitivity when the grayscale levels in the image change significantly. Environments with unstable brightness often make it difficult for recognition algorithms.

- Spatial invariance (algorithm performance): This refers to how well the algorithm performs regardless of where the texture appears in the image.

- Rotational invariance (algorithm performance): This is a characteristic of a recognition or feature extraction algorithm that allows texture analysis systems to operate stably even when the texture is rotated.

- Invariance to projections (2D and 3D projections).

- Algorithm robustness to noise.

- Robustness to different parameter settings.

- Computational complexity: Methods with high computational complexity are often not preferred for practical use.

- Generalization ability of the method: This is also one of the mandatory requirements for texture image classification methods to be applicable in practice. Algorithms that do not meet the required generalization level often perform well on the training dataset but poorly on other datasets.

- The size of the sample image is also a factor that affects the accuracy of texture image analysis systems.

With texture description methods, the performance of a classification method is typically demonstrated by using a texture classification experiment, which usually includes the following steps (it's not always necessary to go through all of them):

- Select image data.

- Divide the image into multiple smaller images to increase the number of images in the training set, because the texture features of an image can be repetitive.

- Pre-process the data.

- Split the image data into a test set and a training set.

- Select a suitable classification algorithm. Choosing the right algorithm is also related to selecting the evaluation method.

- Evaluate the accuracy and performance of the system.

Texture image classification faces challenges similar to those of general image classification. However, a key issue is that if the details of objects within a texture image are not clear enough, it can lead to confusion and ambiguity in the characteristics between different texture classes. Furthermore, if the similarity among images within the same class is not high, the classification process becomes very difficult. Additionally, significant changes in lighting can have a major impact on the results of the image classifiers.

3. LOCAL FEATURE EXTRACTION FROM TEXTURE IMAGES

Texture is used to describe a region where textural elements are characterized by their spatial relationships. An image can consist of one or more textures. If multiple textures exist in an image, the boundaries between them can be detected and distinguished by using predefined texture measures. Texture measures can provide crucial information for image segmentation, feature extraction,

and image classification. Texture measures are very useful in interpreting images from remote sensing satellites, medical magnetic resonance imaging, material science, and aerial terrain imagery. For example, studying urban and rural land development in satellite images can benefit from using image texture analysis.

There are many texture measures to characterize texture. The texture characterization process generates a set of features for the texture in an image. Regional features such as coarseness, uniformity, density, smoothness, straightness, direction, detail, and frequency are often used as texture features. Many approaches, including autocorrelation functions, Gray-Level Co-occurrence Matrices (GLCM), and Local Binary Patterns (LBP), are used to describe and extract texture features in an image. All of these approaches fall into four categories: statistical methods, structural methods, model-based methods, and transform-based methods [9-12]. In most cases, texture features are numerically represented by feature vectors, which include feature components derived from a neighborhood of the corresponding texture class.

Each of these approaches has its own advantages and disadvantages. For example, the statistical approach is suitable for micro-textures (i.e., random textures), whereas the structural approach is a good fit for macro-textures or well-defined texture patterns like periodic textures [13, 14]. Many methods for describing texture characteristics depend on the parameters used, such as the neighborhood size for the texture region, grayscale quantization, and the orientation used to measure relationships between pixels, such as distance and angle [11].

Once texture features are extracted from a texture, the next step is to perform texture classification of the object in the image. Texture image classification methods fall into two main groups: The first group is based on features with high spatial localization. In this group, most edge detection methods can use texture features for spatial localization. The problem with this group of methods is the difficulty in distinguishing between texture boundaries and the edge contours found within the same texture. The second group is based on classification functions with texture features as input. The classification accuracy of this group depends on the capability of the classifier and the suitability of the extracted features for that classifier. Therefore, the most crucial step in this second group is feature extraction, as the extracted features must be able to differentiate between different texture classes [15, 16]. The authors in papers [9-11] define a primitive as a set of connected pixels characterized by a list of attributes. Each primitive is called a texture primitive and can be referred to as a "texel" or "texton." A texture pattern can be described by one or more primitives that are spatially related. The smallest primitive can be the pixels themselves. The characteristics of a primitive are distributed in its neighborhoods. These concepts are widely used and have proven highly effective in texture classification problems for objects in images [17-20]. One of the most famous and widely used methods for texture classification is the Local Binary Pattern (LBP). The LBP method only considers spatial relationships, not the color relationships of pixels.

Today, color images have become one of the most common ways to store information in various industries, media, and information technology. Color features are also a strong potential characteristic for classifying the texture of objects in an image. Furthermore, different objects have different color characteristics, so many methods use color features to classify the texture of objects. Color features are combined with the spatial features of the texture to classify the texture of an object. LBP features and features based on the color distribution of pixels are combined for the problem of classifying the structure of color images. Each texture group has a characteristic pattern that can distinguish it from other groups in a classification task. To determine which texture group a pixel belongs to, its neighboring pixels are analyzed to measure its similarity with a texture and color spectrum group.

3.1. Local Binary Patterns (LBP)

Although global features of texture images have achieved promising results by exploiting the general distribution of pixel values in extremely small neighborhoods (e.g., 3x3), local image feature extraction methods provide a robust texture analysis of pixel intensity values in local neighborhoods. Prominent examples include the Local Binary Pattern (LBP) and its variants.

By computing histograms of local descriptors, LBP-like approaches combine structural and statistical methods, contributing significantly to performance improvements in texture image analysis. The histogram is a sparse vector that summarizes the occurrence of local descriptors at every pixel location, regardless of their spatial position, thus eliminating global image shape and layout.

The LBP was originally derived from the idea of combining local structure analysis from structural methods and occurrence analysis from statistical methods, such as the Gray Level Co-occurrence Matrix (GLCM). The LBP was initially introduced with the basic idea of summarizing a texture region or an image by comparing each pixel to its neighbors (originally a 3x3 neighborhood). For each pixel, a binary code is computed by thresholding the neighboring pixels based on the central pixel's value, as shown in Figure 3. The resulting histogram is then calculated, leading to a 256-dimensional feature vector (2^8=256 possible codes). This descriptor is very popular due to its simplicity, ease of implementation, low computational cost, and its invariance to monotonic changes in illumination.

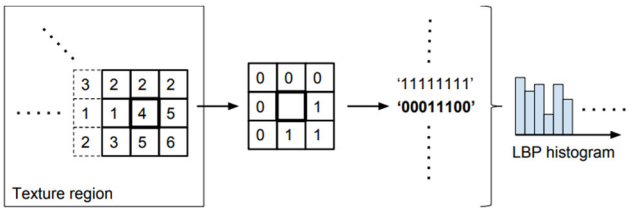

Figure 3. The process of extracting a histogram from the LBP descriptor

However, LBP has several major drawbacks: First, its histogram is sensitive to image rotation. Second, it has a small spatial support (a 3x3 neighborhood), which fails to extract large-scale texture information. Third, LBP loses local texture information (e.g., contrast) by only considering the sign of the difference of neighboring pixels. Finally, it's very sensitive to noise and blurring, as small fluctuations above or below the central value can change the binary code in a way similar to a contrast change. A large number of variations of the original LBP have been proposed to overcome these disadvantages.

A rotation-invariant version named LBPROT is implemented by cyclically rotating the same binary codes. This leads to a reduction in the size of the histogram, which represents the occurrences of 36 unique rotation-invariant patterns. However, the angular space quantization with eight pixels in a circular (square) asymmetric neighborhood is not applied to the calculation of rotational invariance. Moreover, the occurrence of the 36 unique rotation-invariant binary patterns varies greatly because some patterns are unlikely to occur. Rotation-invariant LBP was first improved by using bilinear interpolated intensity values sampled on a circle with a varying radius around the central pixel. The quantization of the angular space can be modified by changing the number of interpolated values, and the local structure can be described at multiple scales by varying the radius. The LBP at a pixel location (x_c,y_c) is calculated using formula (1):

$$LBP_{P,R} = \sum_{p=1}^{P-1} s(I_p - I_c)2^p, s(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

In there, $I_c = I(x_c, y_c)$, P is the interpolated value considered in the neighborhood and R is the radius of the circle on which these values lie.

3.2. Local Directional ZigZag Pattern (LDZP)

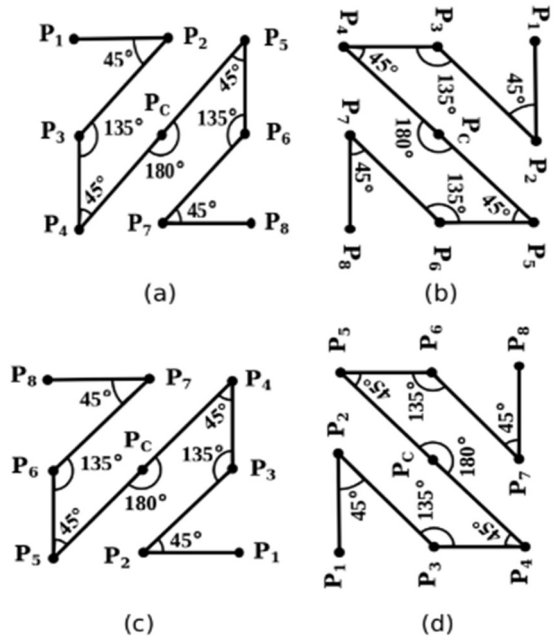

Figure 4. The ZigZag structures

ZigZag LBP effectively represents the characteristic relationship between a central pixel and its neighboring pixels, following a pixel scanning mechanism in a ZigZag pattern known as the ZigZag model. Local ZigZag Pattern (LZP) is a local grayscale texture feature descriptor that represents the local spatial ZigZag structures of texture images, as shown in Figure 4. The input grayscale image is denoted as I, P_c(i, j) is the central pixel of the 3x3 window with a grayscale value I_c(i, j) and the n-th neighboring pixel of P_c(i, j) is denoted as P_n(i, j) with a grayscale value of I_n(i, j), n is a positive integer in the range of [1,N]. The value of N in this case is 8, representing the 8 neighboring pixels of the central pixel. The local texture of a monochrome image I is expressed by the joint distribution of the grayscale value differences between the central pixel and its N neighbors (N>0), which is determined by formula (2):

$$T^{(i,j)} = \tau(I_1^{(i,j)} - I_c^{(i,j)}, I_2^{(i,j)} - I_c^{(i,j)}, I_N^{(i,j)} - I_c^{(i,j)}) \quad (2)$$

Where the function τ(·) represents the joint distribution function. To encode structural information using ZigZag patterns, we only need to consider the sign of the difference between the grayscale values of the central pixel and the neighboring pixel, which is given by $sign(I_n(i,j)-I_c(i,j))$. This is what makes LZP invariant to variations in grayscale values. Therefore, the LZP operator is optimal against changes in illumination intensity and is defined by formula (3):

$$Lzp_{(i,j)} = \sum_{n=1}^{N} sign(I_n^{(i,j)} - I_c^{(i,j)}) \times 2^{n-1}$$
$$sign(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{else} \end{cases} \quad (3)$$

The sign of the difference between the grayscale value of the central pixel and the neighboring pixels is described by a binary string of length N bits, where 2^{n-1} represents the weight of the n-th bit, resulting in 2^N different values for the LZP code. Because the window commonly used to calculate the LZP value is 3x3, each central pixel will have 8 neighboring pixels, and the LZP code value for each pixel will be in the range of 0 - 255. Figure 5 illustrates the ZigZag structure of a 3x3 window, showing the pixel representation based on the ZigZag structure, the weights of the local ZigZag pattern of the window, the original texture pattern, and the LZP pattern of the texture sample. After the LZP codes for each pixel in an $M_x \times M_y$ image are calculated using the formula above, the distribution of the grayscale texture patterns is represented by constructing a discrete 256-column distribution of the calculated LZP codes, as per formula (4).

Where L is the maximum value of the LZP pattern. The LZP feature descriptor is extended to a uniform pattern, where the uniformity measure U encodes the number of bitwise transitions from 0 to 1 and from 1 to 0 in an N-bit pattern, and is defined as follows:

$$U(Lzp_N) = \left| s(I_N^{i,j} - I_c^{i,j}) - s(I_1^{i,j} - I_c^{i,j}) \right| + \sum_{n=2}^{N} \left| s(I_n^{i,j} - I_c^{i,j}) - s(I_{n-1}^{i,j} - I_c^{i,j}) \right| \quad (4)$$

For example, the U value for the LZP strings 11111111 and 00000010 are 0 and 2, respectively. A uniform LZP pattern refers to the uniform occurrence of a string with a limited number of transitions (U ≤ 2) in the circular N-bit binary string. All non-uniform N-bit binary strings (U > 2) are grouped into an "Other" category. The mapping from LZP_N to LZP_N^{u2}, where the superscript "u2" indicates uniform patterns with a U value of at most 2, results in a total of N×(N−1)+3 distinct labels. The calculation of LZP^{u2} is performed using a lookup table of 2^N separate elements. It has been observed that the uniform representation of LZP is more stable (less sensitive to noise), and the number of columns becomes significantly smaller when computing the histogram of LZP codes, which makes matching computationally efficient.

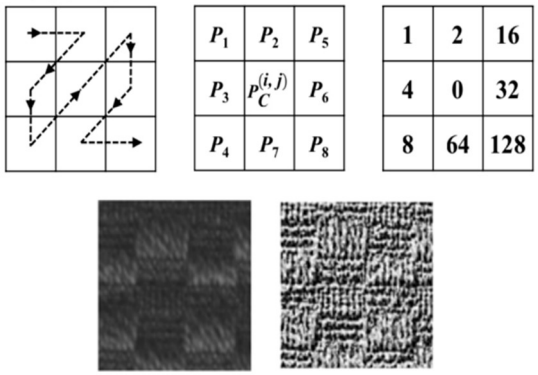

Figure 5. The ZigZag structure of a 3x3 window and image encoding

The images after the texture image encoding process using the Local ZigZag Pattern method on the KTP-TIPS texture image dataset are shown in Figure 6 below, which presents some images randomly selected from the encoded image set:

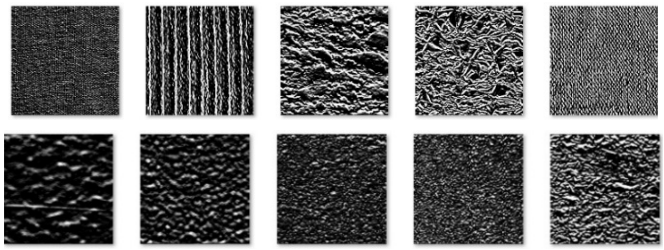

Figure 6. Encoded images using the ZigZag pattern method from the KTH-TIPS texture dataset

4. EXPERIMENTAL RESULTS

Hyperparameter tuning is an essential step in the deep learning model building process, aimed at identifying optimal hyperparameters to improve model performance on the validation set while also ensuring generalization to new data. In this report, a hyperparameter tuning technique has been applied to a deep learning algorithm to perform an image classification task, contributing to improved computational efficiency and enhanced accuracy in data classification.

The classification accuracy of the training method on the texture image dataset, which was encoded using the Zigzag pattern method with hyperparameter tuning in a

deep learning algorithm, reached 99.97%. This is a very high accuracy in texture image classification compared to other methods.

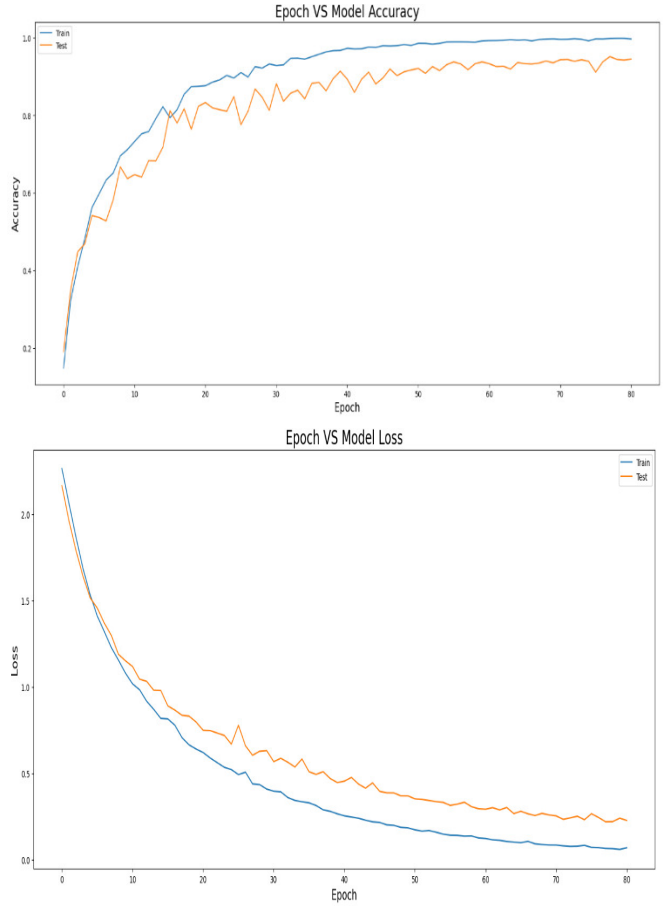

Figure 7. Training model results

Below is a table comparing the results of other models with the experimental model on the KTH-TIPS and Kyberge-28 datasets:

Table 1. Comparison of results between methods on the KTH-TIPS and Kyberge-28 datasets

No	Method	KTH-TIPS	Kyberge-28	Reference
1	Histogram + SVM	59.62%	57.84%	-
2	LBPH + SVM	77.5%	74.5%	-
3	LDZP + SVM	84.13%	81.6%	-
4	MobileNetV1	98.14%	92%	[21]
5	VGG16	98.89%	97.52%	[22]
6	RestNet50	98.51%	97.3%	[23]
7	LDZP + Mobilenet	98,04%	96%	[24]
8	LDZP + VGG16	98,31%	97.26%	[25]
9	LDZP + RestNet50	99.66%	99.21%	[26]
10	**LDZP + Deep Learning**	**99.97%**	**99.54%**	**Proposal**

After comparing the results, it is clear that the LDZP feature extraction method combined with a deep learning method that includes hyperparameter tuning yields superior results compared to other methods. It achieves a nearly perfect recognition accuracy, which is an outstanding result with the potential for effective application in texture recognition tasks that demand high precision.

5. DISCUSSION

The research results demonstrate the superiority of the Local Directional ZigZag Pattern (LDZP) feature extraction method, especially when combined with hyperparameter tuning in deep learning. The accuracy reached 99.97% on the KTH-TIPS dataset, which is significantly higher than many traditional methods and other popular deep learning models. The method's key advantages are its invariance to illumination and noise, thanks to the ZigZag encoding mechanism. However, a drawback of this method is its high computational cost, as the hyperparameter tuning technique requires substantial resources. In the future, we need to expand testing to other types of data and optimize the model to reduce the method's complexity, paving the way for its application in classifying complex textures.

REFERENCES

[1]. J.C. Russ, *The Image Processing Handbook*, 3rd edition. CRC Press, Florida, 1999.

[2]. IEEE Standard 610.4-1990.

[3]. R.M. Haralick, K. Shanmugam, I. Dinstein, "Textural Features for Image Classification," *IEEE Trans. Syst. Man Cybern*., 3, 610-621, 1973.

[4]. R.C. Gonzalez, R.E. Woods, *Digital Image Processing*. Addison-Wesley, Reading, MA, 1992.

[5]. A. Sarkar, K.M.S. Sharma, R.V. Sonak, "A new approach for subset 2-D AR model identification for describing textures," *IEEE Trans. Image Process*., 6, 407-413, 1997.

[6]. G. Cross, A. Jain, "Markov random field texture models," *IEEE Trans. Pattern Anal. Mach. Intell*., 5, 25-39, 1983.

[7]. U. Indhal, T. Næs, "Evaluation of alternative spectral feature extraction methods of textural images for multivariate modeling," *J. Chemometr.*, 12, 261-278, 1998.

[8]. Cimpoi M., Maji S., Kokkinos I., Mohamed S., Vedaldi, A., "Describing textures in the wild," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 3606-3613, 2014.

[9]. Haralick RM, "Statistical and structural approaches to texture," in *Proceedings of IEEE*, 67, 5, 786-804, 1979.

[10]. Haralick RM, Bosley R, *Texture features for image classification*. in: Third ERTS symposium, NASA SP-351, 1219-1228, 1973.

[11]. Haralick RM, Shapiro, *Computer and robot vision*, vol 1. Addison Wesley, Boston, 1992.

[12]. Tso B, Mather P, *Classification methods of remotely sensed data,* 2nd edn. CRC Press, Boca Raton, 2009.

[13]. Petrou M, Sevilla PG, *Image processing: dealing with texture*. Wiley, New York, 2006.

[14]. Pietikainen MK, *Texture analysis in machine vision. Series in machine perception and artificial intelligence*, vol 40. World Scientific, Singapore, 2000.

[15]. Dubes RC, Jain AK, "Random field models in image analysis," *J Appl Stat*, 16:131-164, 1989.

[16]. Frankot RT, Chellapa R, "Lognormal random-field models and their applications to radar image synthesis," *IEEE Trans Geosci Remote Sens* GE-25(2), 1987.

[17]. He DC, Wang L, "Texture unit, texture spectrum, and texture analysis," in: *Proceedings of IGARSS' 89/12th Canadian symposium remote sensing*, 5, 2769-2772, 1989.

[18]. He D-C, Wang L, "Texture unit, texture spectrum, and texture analysis*," IEEE Trans Geosci Remote Sens*, 28(4), 1990.

[19]. Maenpaa T, *The local binary pattern approach to texture analysis - extensions and applications*. Oulun Yliopisto, Oulu, 2003.

[20]. Ojala T, Pietikainen M, Maenpaa T, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Trans Pattern Recognit Mach Intell,* 24 (7), 2002.

[21]. Howard, et al., "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," *arXiv preprint arXiv:1704.04861,* 2017.

[22]. Karen Simonyan, Andrew Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[23]. He, K., Zhan, et al., "Deep Residual Learning for Image Recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (CVPR), 2016.

[24]. Sandler M., Howard A., Zhu M., Zhmoginov A., Chen, L. C., "MobileNetV2: Inverted Residuals and Linear Bottlenecks," in *MobiProceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* 4510-4520, 2018.

[25]. Chollet F., "Xception: Deep learning with depthwise separable," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* 2017.

[26]. Zhang X., Zhou X., et al., "ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* 6848-6856, 2018.