Texture classification model based on adaptive local ternary pattern

Kithi Ngombo[1]
Raphael Angulu[2]
Dorothy Rambim[3]

[1]sifangombo254@gmail.com
[2]rangulu@mmust.ac.ke
[3]drambim@mmust.ac.ke

[1,2,3] Masinde Muliro University of Science and Technology, Kenya

https://doi.org/10.51867/ajernet.7.3.87

ABSTRACT

The success of image processing applications heavily relies on accurate texture analysis. Local Binary Patterns (LBP) is one of the texture models that has found its way into huge application in image texture analysis due to its simplicity and efficiency. However, LBP is more sensitive to noise and can easily classify different pattern into the same class therefore reducing its discriminative property. Local Ternary Pattern was proposed as an extension of LBP and was found to be more resistant to noise. However, LTP and its variants use a static threshold that lacks statistical relationship with the pixel values of an image and which makes it dynamically inappropriate to all images of a dataset or different datasets. This research proposes Adaptive Local Ternary Pattern (ALTP) that uses a dynamically calculated threshold to determine texture pattern for an image region. Experiments conducted on complex texture datasets; (KTHTIPS2b, Stex and Describable Textures Dataset (DTD)), proved ALTP to be more robust than LBP while matching or exceeding the static-threshold LTP with high stability. ALTP's ability to be insensitive to noise and its dynamic threshold coupled with its better performance makes it a great breakthrough in the search for a better texture descriptor. ALTP is recommended as a lightweight, robust alternative for real-world visual inspection, material inspection and edge computing and in the implementation of machine learning image processing pipelines operating under varying surface contrast and noise conditions. Machine Learning engineers building non-linear models that demand high computational efficiency while maintaining stable feature representation will significantly benefit from the power of ALTP.

Keywords: Adaptive Local Ternary Pattern, Feature Extraction, Local Binary Pattern, Local Ternary Pattern, Texture Classification

I. INTRODUCTION

Texture is a simple and yet a complex term and topic in image processing. It can be loosely defined as the tangible feel of a surface or its visual appearance in terms of brightness, shape or contents. The human visual system is well adapted to distinguish texture with ease; however, this is not the case when it comes to computer image processing. To achieve texture analysis, texture feature extraction is necessary. This is the process of quantifying texture characteristics in terms of descriptors and quantitative feature measurements. The set of these features form a feature vector which the computer uses to achieve image analysis. Texture is therefore, a highly valuable feature in image processing, primarily serving four main purposes: classification, synthesis, segmentation, and shape determination (Armi & Fekri-Ershad, 2019).

Texture classification deals with categorizing textural images into different categories (Armi & Fekri-Ershad, 2019). The aim of this is to assign an unknown image into one of the known categories such categorizing a new face image as either a male or female face, or a mammogram as either malignant or benign. The texture classification process goes through two phases; learning and recognition phases. Learning involves building a model where a set of training images with known class labels are used. The objective here is for the model to be able to extract the texture properties that uniquely distinguish the images of a given texture class from another. Since many very sophisticated classifiers exist, the key challenge here is the development of effective features to extract from a given textured image.

Texture synthesis is mainly used to produce images similar to the input images, a technique heavily used in computer games and creation of graphic images, scene creation and artistic effects on images. The goal of texture synthesis is to create an output image that; (1) have the size specified by the user, (2) is similar to the sample image, (3) does not have visible artifacts such as edge misfitting, and blocks and (4) does not have repeats where the same structure in the output image appears in multiple places. Texture synthesis can be achieved using such algorithms as Tiling (Toulatzis & Fudos, 2022), Stochastic texture synthesis (Gagalowicz et al., 1981), Single purpose structured texture

synthesis, Chaos mosaic (Guo et al., 2000), Pixel-based texture synthesis (Longstaff & Rupert, 1998), Patch-based texture synthesis (Alexei & William, 2001) and Deep Learning and Neural Network Approaches (Leon et al., 2015).

Texture segmentation focuses on dividing an image into different regions each of which have different texture. This property is heavily used in medical digital image processing like detection of different types of cancer where the cancerous tissues are segmented from the uninfected tissues in an image due to the different textual properties among many other application areas. Finally, texture shape extraction is mainly used in 3D image extraction from images of cause easily distinguished by their different textures (Armi & Fekri-Ershad, 2019). Despite the vast expanse of the application areas of texture analysis, the area is faced with four major challenges; rotation, noise, scale and illumination intensity (Ramola et al., 2020). Texture analysis can be greatly affected be it for the purpose of classification, synthesis, segmentation or shape extraction if the method used is not robust enough to deal with these challenges. This explains why a lot of research is still underway to define robust texture descriptors both in theory and practice to address the aforementioned challenges.

Texture analysis is therefore a fundamental process in today's computer vision and image processing (Mwadulo et al., 2020). Texture extraction is useful in fields such as pattern recognition, segmentation, medical image analysis, remote sensing, object detection, product quality diagnostics and texture classification (Manish et al., 2004). The information obtained from this process of texture feature extraction is used in remote sensing, biomedical imaging, and inspection. According to Tuceryan and Jain (1998), texture analysis methods can be categorized into statistical methods, geometric methods, model-based and signal processing methods.

Statistical methods mainly focus on the spatial distribution of pixel values (Ramola et al., 2020). Based on this, statistical methods are further sub-classified as first-order statistics, second-order statistics, and high-order statistics. First-order statistics includes mean, variance, standard deviation, kurtosis, and skewness. This classification does not provide sufficient information with respect to human visual perception. On the other hand, second-order statistics explores the relationship between a pixel of interest (POI) and its neighborhood pixels (Lukashevich & Sadykhov, 2012). It includes Gray Level Co-occurrence Matrix (GLCM), Autocorrelation function, Local Binary Pattern (LBP) and Local Ternary Patterns (LTP) and their variants. Second-order statistics provide sufficient information with respect to human visual perception thus making the methods in second-order statistics popular for use in describing texture. High-order statistics do not provide any spectral or spatial information which is critical in describing textures thus they are not used in image interpretation (Laleh & Fekri-Ershad, 2019).

Geometric texture analysis defines texture based on spatial layout and initial units which can be a pixel, a region or a line. The spatial layout rules of these units are drawn by calculating the geometric relationships or examining their statistical properties. The patterns of the identified initial units then define the texture of a given image. Scale Invariant Feature Transform (SIFT) (Lowe, 1999) is a good example of geometric texture analysis methods. Model-based texture analysis methods describe texture by comparing a given texture with predefined standard model design. Autoregressive model, Hidden Markov model, Fractal model, Gibbs random field and Markov square theory are some of the popular models used for texture analysis (Ramola et al., 2020). Finally, transform-based methods use transformation functions to convert a given image into another form. This means that the success of these methods is determined by the transformation function used. Examples of techniques widely used under this category are Fast Fourier Transform (FFT), Multi-way Principal Component Analysis (MPCA), Wavelet Transforms, Gabor and Rigdelet (Manish et al., 2004).

The increased application of image processing and computer vision in organizations/industries means that more efficient and robust feature descriptors are needed to perform real-time tasks with utmost efficiency and accuracy. This is because the successful classification of any algorithm in a model heavily relies on the reliability of the extracted features passed to it. Incorrect features will definitely lead to incorrect classification even with a very well performing model (Phuong et al., 2015). This explains why a lot of research has been focused on coming up with feature descriptors that will be more efficient yet less complex. Local Binary Patterns (LBP), introduced in the early 1990's emerged as a very promising local feature descriptor due to its computational simplicity and discriminative power. Its success was demonstrated by its huge application in surveillance and security and biomedical imaging.

Local Ternary Pattern (LTP) is just but one of those texture feature extractors that have drown much interest from researchers and engineers from across the globe interested in this promising field. LTP was a very important breakthrough in local texture feature extraction that saw growth in texture feature extraction in a wide range of application (Xiaoyang & Bill, 2010). Its introduction which was mainly to address the limitations of LBP such as sensitivity to noise attracted more interest from researchers in the recent years. The desire by researchers to create an even more efficient feature descriptor led to the introduction of many variants of LTP in an effort to address the weaknesses that were notable in LTP.

Though LTP is much efficient than LBP, it is not strictly invariant to gray level transformation due to the simple yet ambiguous strategy of selecting a threshold (Jing-Hua et al., 2014). This is because the choice of a threshold is left to the user to pick any number to use as threshold, whereby the chosen threshold may not be optimum for all images in a dataset. Moreover, identifying a suitable threshold is a challenge since the room to choose from is not confined to a

given range of values hence making it ambiguous and quite time consuming in selecting an optimum threshold (Angulu et al., 2018). Local Ternary Pattern also does not pay attention to the possibility of the presence of outliers, a weakness that has not been fully addressed in subsequent variants. When outliers are allowed to contribute to the resultant output of every local patch of an image processing the resultant feature vector becomes inaccurate as there is a buildup of erroneous features.

1.1 Statement of the Problem

Feature extraction is a key aspect in machine learning that determines the overall performance of machine learning algorithms and the texture of an image is critical in this case. Finding a more robust feature extractor is therefore crucial to achieve better results in machine learning. The LTP, an extension of LBP was well accepted as it dealt with some of the weaknesses of LBP. However, a number of researchers - as will be discussed in the next topic - have pointed out that the static threshold in LTP is a challenge since the threshold does not adapt to the local image statistics across images in a given dataset and that the choice of an optimum threshold value is difficult since there is no defined approach to obtaining a threshold. This research is aimed at finding an appropriate way of obtaining a threshold that will be variant to the grey values of images and with a well-defined way of obtaining the threshold with a focus on statistical data distribution analysis techniques such as standard deviation, median and interquartile range.

1.2 Research Objective

To develop a machine learning model for texture classification using adaptive local ternary pattern.

II. LITERATURE REVIEW

Texture classification aims at assigning an unknown sample image to one of a set of known texture classes (Ojala & Pietikainen). Color and texture are two siblings that form the foundational basis for computer vision and image processing as they take the front line in human visual perception (Humeau-Heurtier, 2019). Though it is pretty easy for the human eye to distinguish images based on their color or texture Francesco et al. (2021) noted that the urge to replicate this ability to computers in Artificial intelligence has led to ever growing research works in the field of computer vision and image processing. Texture forms the fundamental aspect of an image that contributes to its recognition (Goyal & Sharma, 2022). In (Philomina & Uma, 2020) Philomina and Uma present deep learning approaches in performing feature extraction for texture classification. Feature extraction, a major path to texture classification can be approached in two broad ways: (1) Local feature extraction and (2) Global feature extraction. While local feature extraction focuses on subdividing a given image into small region for analysis, global feature extraction analyses a given image as a whole. With focus on local feature descriptor, there are a number of approaches used in feature extraction some of which are discussed below.

Local Binary Pattern

Local Binary Pattern (LBP) is a local feature descriptor that revolutionized the field of image processing after its introduction. The focus of LBP was on the analysis of rotation invariant texture classification based on gray scale images (Ojala et al., 2002). LBP achieves this by binarizing every 3 x 3 local patches of the image by comparing the central pixel against the neighboring pixel intensities. The resulting encoding of the image then becomes the features of the texture of the image. LBP however is not rotation invariant and it is sensitive to noise. To address rotation invariance in LBP a rotation invariant LBP was proposed (Ojala et al., 2002).

Local Ternary Pattern

Local Ternary Pattern was introduced by Tan and Triggs, (Xiaoyang & Bill, 2010) as an extension of LBP – a 2 value (binary) code – to a 3-value code by introducing a threshold t in which gray-levels within $\pm t$ around the central pixel c_i are quantized to 0, those above $i_c + t$ are quantized to +1 and those below $i_c - t$ are quantized to -1 using the function (1). This threshold in LTP is manually set in a try and error method until the best performance is achieved.

$$s'(u, i_c, t) = \begin{cases} 1, & u \geq i_c + t \\ 0, & |u - i_c| \leq t \\ -1, & u \leq i_c - t \end{cases} \quad (1)$$

The introduction of a threshold in LTP made it more resistant to noise but not strictly invariant to gray-scale changes (Isaam et al., 2018). This results into an 8-bit ternary pattern which is then split into two channels; positive and negative channels.

The binary pattern for the positive pattern is calculated as;

$$u_i = \begin{cases} 1, & if\ i = 1 \\ 0, & otherwise \end{cases} \quad (2)$$

The binary pattern for the negative pattern is calculated as;

$$l_i = \begin{cases} 1, & if\ i = 1 \\ 0, & otherwise \end{cases} \quad (3)$$

These two channels are used to create separate histogram, compute similarity matrices and are then combined once the computation is done. The introduction of ternary code in LTP gave it an advantage over its predecessor, LBP. However, the static threshold in LTP does not adapt to the local image statistics in a given dataset and that the choice of an optimum threshold value is difficult since there is no defined approach to obtaining a threshold.

Local Directional Ternary Pattern

Based on LTP and LDP as the foundation, Local Directional Ternary Pattern LDTP was proposed in 2018 mainly for facial expression recognition (Isaam et al., 2018). LDTP applies Kirsch masks to compute responses for all the eight directions for central reference pixel in a 3 x 3 image region. Ever since the introduction of LTP, many research works have been inspired by LTP in the quest for local feature descriptors. Mary Mwadulo et al. (Mwadulo et al., 2020) inspired by LTP, Proposed Local Direction Ternary Pattern for breast cancer classification a descriptor that displayed great competence in performance.

III. METHODOLOGY

The research adopted a simulation and experimental strategy following the general machine learning research design presented by Kamiri and Mariga (Kamiri & Mariga, 2021), though with slight modification as presented in **figure 1** below.

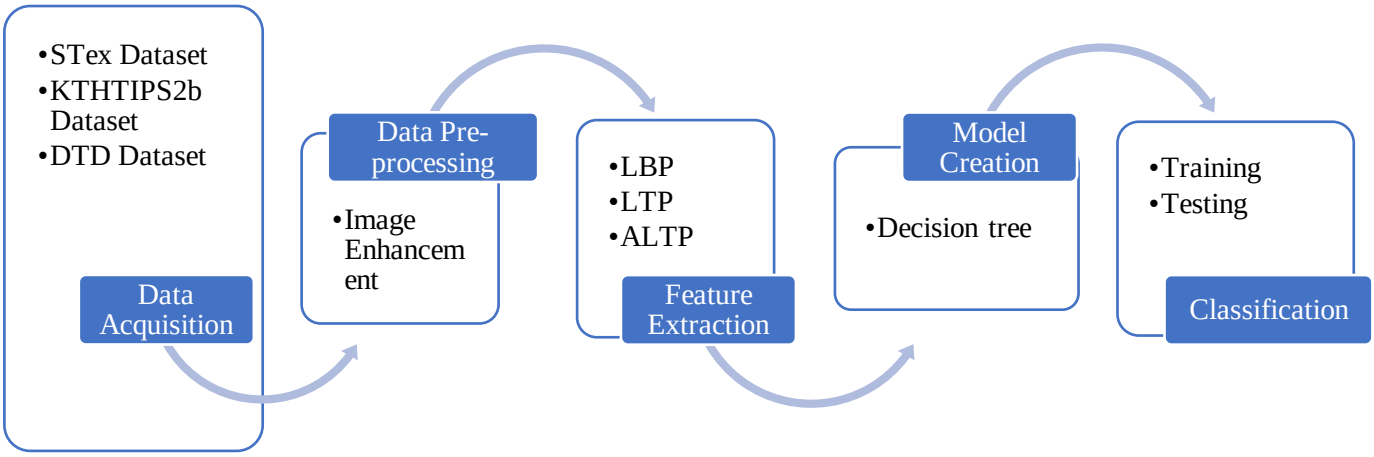

Figure 1

Experimental pipeline of the research methodology adopted from (Kamiri & Mariga, 2021)

The simulation was set up on an HP EliteBook Laptop with Windows 11 Pro (64-bit), Intel Core i7 processor, 2.6 GHz, 8 GB RAM. Python 3.12.3, Jupyter notebook, Panda 2.0.2, NumPy, sklearn 1.6.1 and matplotlib 3.10.0.

3.1 Data Acquisition

This research used three (3) public datasets: the KTH-TIPS2b database (Mallikarjuna, et al., 2006) is a multiclass dataset that contains images from 11 materials, each consisting 432 images. The images in every class come from 4 different samples (with 108 images per sample) each under varying illumination, pose and scale. The STex database (short WaveLab, n.d.) contains 3 different sized categories of colour texture images; stex-1024, stex-512 and stex-512-splitted of size 1024x1024, 512x512 and 128x128 pixels respectively and the images are fairly homogenous and Describable Textures Dataset (DTD) (Mircea et al., 2014) which consists of 5640 images grouped into 47 classes based on human perception with each class containing 120 images of varying sizes. The databases were chosen due to their complexity in terms of varying illumination, pose and scale which in turn make them suitable for solving texture classification problems.

3.2 Data pre-processing

This is a critical phase in image processing used to improve image quality for better results in subsequent steps. This stage involved contrast enhancement and resizing. To achieve enhancement in terms of image contrast, CLAHE technique was used due to its efficiency. Image resizing was necessary so as to reduce processing time since large images result in large dimension of the feature dataset. Resizing the input image to a smaller size reduces the feature vector thus enhancing future processing. In this study, all input images were resized to 50 x 50 using the OpenCV image resize command.

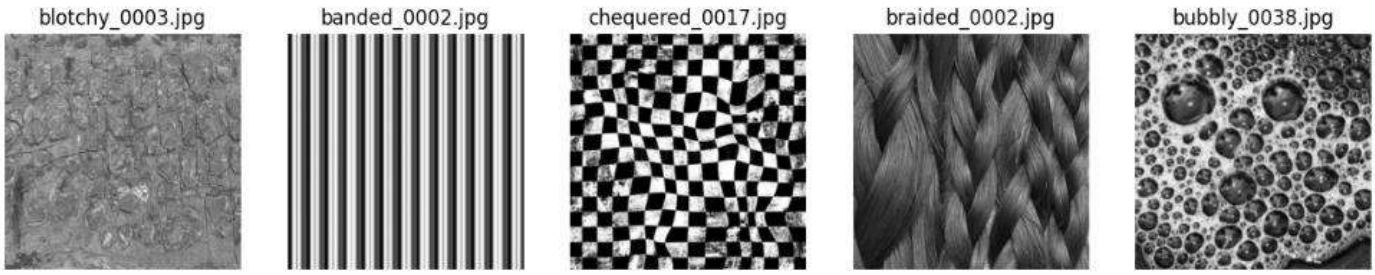

Figure 2
A sample of DTD Dataset resized images

3.3 Feature Extraction

The pre-processed image from the previous phase becomes the input of this phase. This is where the proposed ALTP texture descriptor is used alongside LTP and LBP.

3.4 Adaptive Local Ternary Pattern

Local Ternary Pattern has shown great success as a local feature descriptor which through its 3-valued encoding scheme, it eliminated the issue of noise sensitivity previously experienced in LBP. This was achieved through the introduction of a threshold $\boldsymbol{t}$ that indicated a range around the central pixel $\boldsymbol{i_c}$ within which pixel intensities would be assigned a value **0**, while pixel intensities above that range would be assigned a value **+1** and those below a **-1** value. However, the threshold proposed in LTP is static and open to the user's choice. This gives rise to two main challenges; (1) a threshold that does not adapt to the local pixel intensities of different patches across all images of a given dataset and (2) a threshold that makes it difficult for the user of the algorithm to select an appropriate threshold since there is no specific range within which the threshold can be picked. This section proposes the ALTP thresholding mechanism while still embracing the 3-value encoding scheme as proposed in the original LTP texture descriptor.

Given a 3×3 image region with pixel values $x_1, x_2, \ldots x_n$, where $n = 9$, ALTP sorts the pixel values in ascending order in a set op, and calculates the index, i, of the median pixel as

$$i = \left\{\frac{n+1}{2}\right\}$$

where n is number of pixels in the ordered set op.

The median pixel mp, is picked as

$$mp = f(i) \qquad (4)$$

where $f(i)$ is a function that returns the pixel at index i in the ordered set op. Median Absolute Deviation (MAD) is calculated as

$$MAD = \frac{\sum_{i=1}^{n} |x_i - mp|}{n} \qquad (5)$$

Then, the variance of the pixel values from MAD is calculated as

$$MAD_v = \frac{\sum_{i=1}^{n} (x_i - MAD)^2}{n} \qquad (6)$$

Standard Median Absolute Deviation $\boldsymbol{\theta}$ is derived as

$$\boldsymbol{\theta} = \sqrt{\frac{\sum_{i=1}^{n} (x_i - MAD)^2}{n}} \qquad (7)$$

and the adaptive threshold $\boldsymbol{t}$ is calculated as

$$\boldsymbol{t} = {}^{\boldsymbol{\theta}}/_{k} \qquad (8)$$

where $k = 3$, a constant grounded in sampling theory and spatial uncertainty estimation. In a $3\ x\ 3$ local image patch, the total sample size of intensity values is $N = 9$. According to the Central Limit Theorem, the Standard Error of the Mean (SEM) of the localized mean intensity estimator is expressed as:

$$SEM = \frac{\sigma_\Omega}{\sqrt{N}} = \frac{\sigma_\Omega}{\sqrt{9}} = \frac{\sigma_\Omega}{3} \quad (9)$$

where σ_Ω is the standard deviation around the mean of a given sample.
By borrowing this idea in our median standard deviation θ,

$$SE_{median} = \frac{\theta}{\sqrt{9}} = \frac{\theta}{3} \quad (10)$$

which in our case,

$$SE_{median} = \boldsymbol{t} \quad (11)$$

By using median which has a breakdown point of 50% as opposed to mean which has a breakdown point of 0% and by scaling $k = 3$, our threshold remains insensitive to noise to up to 4 corrupted pixel intensities within the $3\ x\ 3$ neighborhood. This has two advantages; (1) Noise suppression, where variations below t are zeroed out as non-informative background noise (Luping et al., 2018). and (2) Feature preservation, where real micro-texture edges exceeding t are retained and encoded into upper or lower ternary states ($+1$ or -1).
The ALTP code for a neighboring pixel is then computed as

$$S(p_c, p_n, t) = \begin{cases} 1 & if\ p_n \geq p_c + t \\ -1 & if\ p_n \leq p_c - t \\ 0 & otherwise \end{cases} \quad (12)$$

where p_c is the central pixel intensity, p_n is the neighboring pixel intensity of the local patch and $\boldsymbol{t}$ is the adaptive threshold.
The encoding process for ALTP is illustrated in ***Figure 3*** below.

80	78	57
43	54	30
25	54	30

(a)

1	1	0
0		-1
-1	0	-1

(b)

1	1	0
0		0
0	0	0

(c)

0	0	0
0		1
1	0	1

(d)

Figure 3
Resultant Ternary Pattern code with (a) a 3x3 image region, (b) Resultant ALTP code, (c) Positive ALTP code and (d) Negative ALTP code

Ternary code = 110(-1) (-1) (0) (-1)0

The ternary pattern is then split into its positive and negative parts as illustrated in Figure 5 (c) and (d) above. This results into two feature sets named upper – for the positive binary code – and lower – for the negative binary code. The binary pattern for the positive pattern is calculated as;

$$ALTP_u = \begin{cases} 1, & u = 1 \\ 0, & otherwise \end{cases} \quad (13)$$

The binary pattern for the negative pattern is calculated as;

$$ALTP_l = \begin{cases} 1, & i = -1 \\ 0, & otherwise \end{cases} \quad (14)$$

Where $\boldsymbol{\alpha}$ is the ternary code of the neighboring pixel and $\boldsymbol{g} = \{1, 2, \ldots, 8\}$ the indices of the neighboring pixels.

Two histograms for the upper and lower feature sets are then generated which are fused and converted into a one-dimensional array then stored as features for classification.
Figures **4, 5 6, 7** and **8** below show samples of processed images. The samples represent the results of both the upper (positive) pattern and lower (negative) pattern for ALTP and LTP respectively and one sample forLBP.

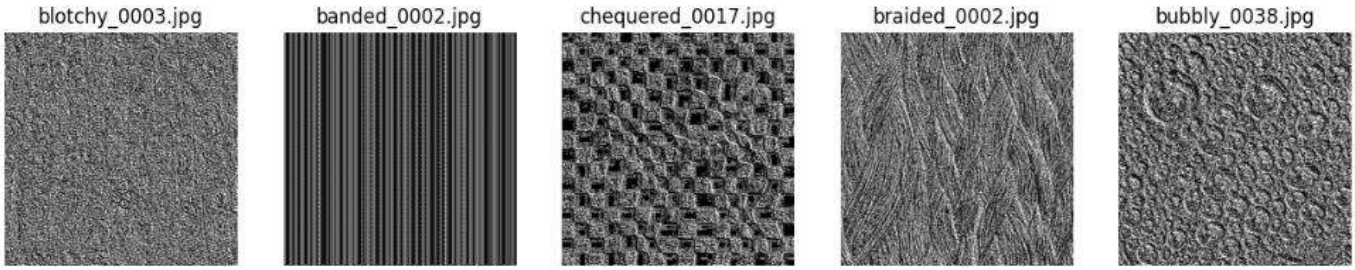

Figure 4
DTD Dataset - ALTP (Upper)

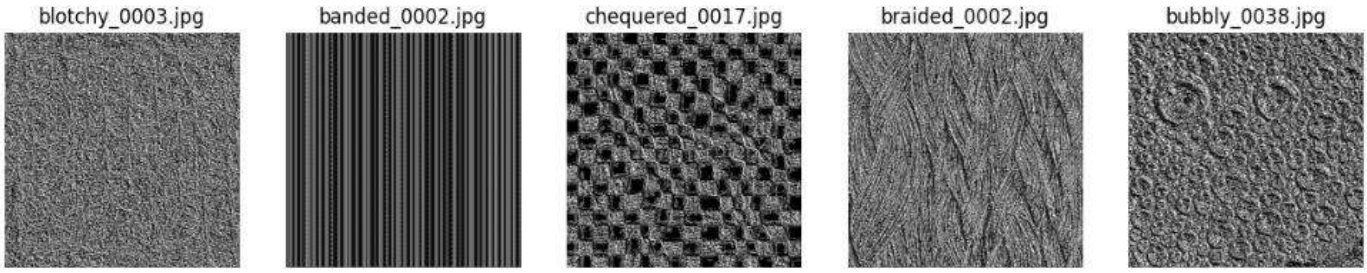

Figure 5
DTD Dataset – ALTP (Lower)

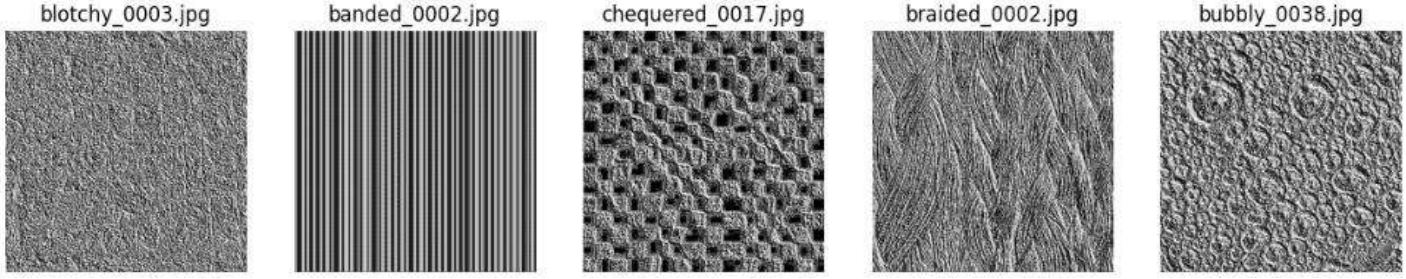

Figure 6
DTD Dataset – LTP (Upper)

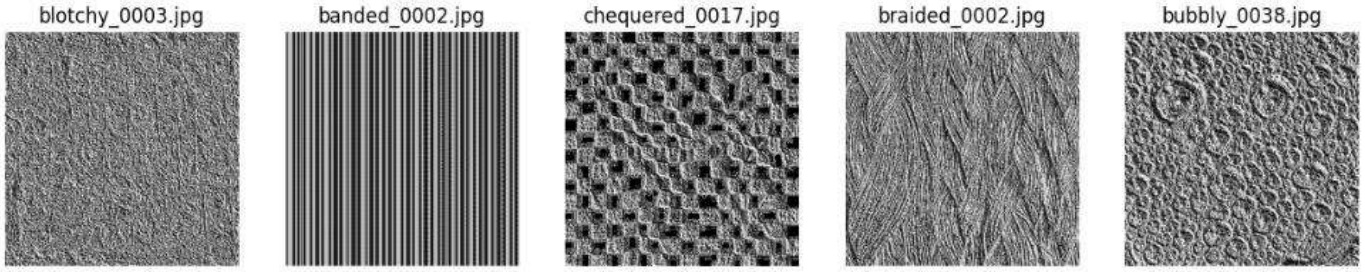

Figure 7
DTD Dataset – LTP (Lower)

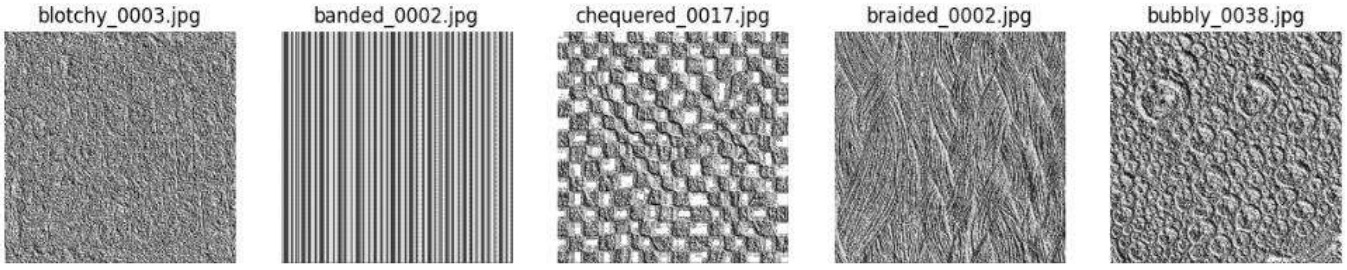

Figure 8
DTD Dataset - LBP

3.5 Classification

This was the final stage of the experimental pipeline. The classification approach used in this study was a multi-class classification approach since all the datasets used for this study more than 10 classes of texture images.

IV. FINDINGS & DISCUSSION

4.1 Experimental Results

To evaluate the proposed method, this research used the previously discussed benchmark texture databases. All input images were subjected to some preprocessing including resizing to 50 x 50, gray-scaling and contrast equalization for better processing. The images were then divided randomly into training, validation and testing sets as shown in **table 1** below. The training set was used for training the classification algorithms used in this research, the testing set was used to test the performance of the trained model while the validation set was used for fine tuning the hyperparameters – number of splits, maximum depth - of the classification algorithms used.

Table 1

Number of images in Train, Validation and Test sets

Dataset	Number of Classes	Training set (70%)	Validation set (20%)	Testing set (10%)
KTHT-IPS2b dataset	11	3326	950	476
STex dataset	32	2083	596	297
DTD dataset	47	3864	1104	552

Based on the databases used in this research which were all multi-class databases, multi-class classification algorithms were used for evaluating the proposed texture descriptor. Three classification algorithms were used on for creating models to evaluate the effectiveness of the proposed feature descriptor ALTP along with LBP and LTP. These algorithms are; Support Vector Machine (SVM), Decision tree, and K-Nearest Neighbor. 10-fold Cross validation accuracy performance alongside a computation of the statistical significance (p-values). The experimental results are presented in the tables below.

Table 2

Cross-validation accuracy (Mean ± SD, 95% CI) and statistical significance (p-values) comparing LBP, LTP, and Proposed ALTP on the KTH-TIPS-2b dataset.

Classifier	Descriptor	Mean ± SD (%)	95% CI (%)	p-value vs ALTP
SVM	LBP	97.37 ± 0.73	[96.85, 97.89]	$p = 0.0020$
SVM	LTP	98.53 ± 0.56	[98.13, 98.93]	$p = 1.0000$
SVM	Proposed ALTP	98.17 ± 0.60	[97.74, 98.60]	$-\ (Baseline$
KNN	LBP	91.36 ± 1.27	[90.49, 92.30]	$p < 0.001$
KNN	LTP	94.70 ± 0.76	[94.16, 95.24]	$p = 0.9863$
KNN	Proposed ALTP	94.04 ± 0.89	[93.41, 94.68]	$-\ (Baseline$
Random Forest	LBP	91.92 ± 0.79	[91.35, 92.49]	$p < 0.001$
Random Forest	LTP	94.87 ± 0.88	[94.23, 95.50]	$p = 0.5781$
Random Forest	Proposed ALTP	94.76 ± 0.64	[94.30, 95.22]	$-\ (Baseline$

Table 3

Cross-validation accuracy (Mean ± SD, 95% CI) and statistical significance (p-values) comparing LBP, LTP, and Proposed ALTP on the STex dataset.

Classifier	Descriptor	Mean ± SD (%)	95% CI (%)	p-value vs ALTP
SVM	LBP	40.94 ± 11.07	[33.02, 48.86]	$p = 0.1035$
SVM	LTP	40.94 ± 8.52	[34.84, 47.03]	$p = 0.1328$
SVM	Proposed ALTP	44.06 ± 6.82	[39.18, 48.94]	$-\ (Baseline$
KNN	LBP	29.38 ± 5.55	[25.40, 33.35]	$p = 0.2754$
KNN	LTP	32.19 ± 6.08	[27.84, 36.54]	$p = 0.7783$
KNN	Proposed ALTP	30.94 ± 8.65	[24.75, 37.12]	$-\ (Baseline$
Random Forest	LBP	37.81 ± 11.73	[29.42, 46.21]	$p = 0.0195$
Random Forest	LTP	38.75 ± 5.93	[34.51, 42.99]	$p = 0.0078$
Random Forest	Proposed ALTP	44.38 ± 6.04	[40.06, 48.68]	$-\ (Baseline$

Table 4

Cross-validation accuracy (Mean ± SD, 95% CI) and statistical significance (p-values) comparing LBP, LTP, and Proposed ALTP on the DTD dataset.

Classifier	Descriptor	Mean ± SD (%)	95% CI (%)	p-value vs ALTP
SVM	LBP	58.92 ± 4.75	[55.52, 62.31]	$p = 0.0332$
SVM	LTP	64.33 ± 5.40	[60.47, 68.19]	$p = 0.9883$
SVM	Proposed ALTP	61.92 ± 4.16	[58.94, 64.89]	$-(Baseline$
KNN	LBP	45.17 ± 4.90	[41.66, 48.67]	$p = 0.0371$
KNN	LTP	48.00 ± 4.97	[44.44, 51.56]	$p = 0.3145$
KNN	Proposed ALTP	48.67 ± 2.55	[46.84, 50.49]	$-(Baseline$
Random Forest	LBP	54.08 ± 5.20	[50.36, 57.81]	$p = 0.0059$
Random Forest	LTP	57.25 ± 6.36	[52.70, 61.80]	$p = 0.0625$
Random Forest	Proposed ALTP	59.75 ± 5.00	[56.17, 63.33]	$-(Baseline$

To evaluate the feature representation capabilities of the proposed ALTP feature descriptor, 10-fold cross-validation experiments were conducted across the three benchmark texture datasets as shown in tables **2**, **3** and **4** above. As can be observed in the results, all three datasets and three ML algorithms, the proposed ALTP consistently outperformed the baseline LBP with strong statistical significance ($p < 0.05$ via paired Wilcoxon signed-rank tests). On DTD, ALTP achieved mean accuracy improvements over LBP of $+3.00\%$ under SVM ($61.92 \pm 4.16\%$ vs. $58.92 \pm 4.75\%$, $p = 0.0332$), $+3.50\%$ under KNN ($48.67 \pm 2.55\%$ vs. $45.17 \pm 4.90\%$, $p = 0.0371$), and $+5.67\%$ under Random Forest ($59.75 \pm 5.00\%$ vs. $54.08 \pm 5.20\%$, $p = 0.0059$). A similar trend emerged on KTH-TIPS-2b, where ALTP demonstrated remarkable performance over LBP ($p < 0.002$ across all classifiers), reaching a peak accuracy of $98.17 \pm 0.60\%$ under SVM. This shows that substituting binary thresholding with an adaptive threshold effectively handles local intensity fluctuations and suppresses background surface noise.

Relative to LTP performance, ALTP was seen to be greatest when applied to textures of high variance and fine-grained micro-textures (STEx). ALTP was able to greatly outperform LTP on the STEx dataset when utilizing a non-linear ensemble classifier (44.38±6.04% vs. 38.75±5.93%, p=0.0078) and linear margin classifier (44.06±6.82% vs. 40.94±8.52%, p=0.1328). On limited variation datasets such as KTH-TIPS-2b, ALTP also demonstrated parity with extensively parameter tuned LTP results (98.17% vs. 98.53% under SVM, 94.76% vs. 94.87% under Random Forest) without the necessity of parameter tuning. Additionally, ALTP displayed significantly lower standard deviations across cross validation folds consistently (e. g. reduced fold deviation from ±4.97% to ±2.55% on DTD using KNN), proving that the proposed ALTP reduces the variance of feature encoding with respect to intra-class texture variation and local image noise.

V. CONCLUSION & RECOMMENDATIONS

5.1 Conclusion

ALTP is a texture descriptor that has proved to be more robust than LBP and LTP. Its ability to be insensitive to noise and its dynamic threshold coupled with its better performance makes it a great breakthrough in the search for a better texture descriptor. The ALTP algorithm only utilized the power of statistical data distribution approaches to come up with a dynamic threshold specifically the absolute median deviation. This means that there is still room for exploration in other areas of possibility where maybe even much better texture descriptors can be derived.

5.2 Recommendations

Based on the experimental results presented here, we strongly recommend adopting the proposed Adaptive Local Ternary Pattern (ALTP) feature descriptor for real-world texture classification task, especially in unconstrained environments where lighting, local contrast and noise levels fluctuate unpredictably. With the elimination of hyperparameter tuning, ALTP provides an autonomous, parameter-free feature extraction pipeline that adapts locally to spatial dispersion. Systems in which computational efficiency and stable feature representation are paramount (like medical image analysis, remote sensing or material surface inspection) will significantly benefit from ALTP's low variance and superior classification accuracy under non-linear models like Random Forest and SVM.

Declaration of Interest

The authors declare that they do not have any known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Funding Declaration

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

REFERENCES

Armi, L., & Fekri-Ershad, S. (2019). Texture image analysis and texture classification methods: A review. *International Online Journal of Image Processing and Pattern Recognition, 2*(1), 1–29.

Bates, S., Hastie, T., & Tibshirani, R. (2023). Cross-validation: What does it estimate and how well does it do it? *Journal of the American Statistical Association, 119*(546), 1434–1445. https://doi.org/10.1080/01621459.2023.2197686

Gagalowicz, A., Faugeras, O., & Pratt, K. (1981). Applications of stochastic texture field models to image processing. *Proceedings of the IEEE, 69*(5), 542–551. https://doi.org/10.1109/PROC.1981.12023

Goyal, V., & Sharma, S. (2022). Texture classification for visual data using transfer learning. *Multimedia Tools and Applications, 82*, 24841–24864. https://doi.org/10.1007/s11042-022-14276-y

Guo, B., Shum, H., & Xu, Y.-Q. (2000). *Chaos mosaic: Fast and memory efficient texture synthesis*. Microsoft Research.

Humeau-Heurtier, A. (2019). Texture feature extraction methods: A survey. *IEEE Access*. https://doi.org/10.1109/ACCESS.2018.2890743

Isaam, E.-K., Abderrazak, C., Youssef, E., & Yassine, R. (2018). Local directional ternary pattern: A new texture descriptor for texture classification. *Computer Vision and Image Understanding, 169*, 14–27.

Jing-Hua, Y., Hao-Dong, Z., Yong, G., & Li, S. (2014). Enhanced local ternary pattern for texture classification. In *Intelligent Computing Theory* (pp. 443–448). Springer. https://doi.org/10.1007/978-3-319-09333-8_48

Kamiri, J., & Mariga, G. (2021). Research methods in machine learning: A content analysis. *International Journal of Computer and Information Technology, 10*, 78–91. https://doi.org/10.24203/ijcit.v10i2.79

Laleh, A., & Fekri-Ershad, S. (2019). Texture image analysis and texture classification methods: A review. *International Online Journal of Image Processing and Pattern Recognition, 2*(1), 1–29.

Leon, G. A., Alexander, S. E., & Bethge, M. (2015). Texture synthesis using convolutional neural networks. In *Advances in Neural Information Processing Systems 28* (pp. 262–270).

Li, L., Lin, F., Sheng-Lan, L., Mu-Xin, S., Jun, W., & Hui-Bing, W. (2018). Intensity-based co-occurrence local ternary patterns for image retrieval. *Journal of Computers, 29*(4), 12–30. https://doi.org/10.3966/199115992018082904002

Longstaff, D., & Rupert, P. (1998). Semi-causal nonparametric Markov random field texture synthesis. *IEEE Transactions on Image Processing, 7*.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision* (Vol. 2, pp. 1150–1157).

Lukashevich, M., & Sadykhov, R. (2012). Texture analysis: Algorithm for texture features computation. In *Proceedings of the International Conference "Problems of Cybernetics and Informatics" (PCI'2012)* (pp. 12–14).

Luping, J., Yan, R., Xiaorong, P., & Guisong, L. (2018). Median local ternary patterns optimized with rotation-invariant uniform-three mapping for noisy texture classification. *Pattern Recognition*, 387–401. https://doi.org/10.1016/j.patcog.2018.02.009

Madasamy, G. R., & V., S. (2013). Optimized local ternary patterns: A new texture model with set of optimal patterns for texture analysis. *Journal of Computer Science, 9*(1), 1–15.

Mallikarjuna, R. P., Alireza, T. T., Mario, F., Eric, H., Barbara, C., & Jan-Olof, E. (2006). *The KTH-TIPS2 image database*. KTH Computer Vision and Pattern Recognition Laboratory. https://www.csc.kth.se/cvap/databases/kth-tips/index.html

Manish, H., Jay, L. J., & John, F. (2004). Image texture analysis: Methods and comparisons. *Chemometrics and Intelligent Laboratory Systems, 72*(1), 57–71.

Mircea, C., Subhransu, M., Iasonas, K., Sammy, M., & Andrea, V. (2014). Describing textures in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 3606–3613). https://doi.org/10.1109/CVPR.2014.461

Mwadulo, M., Mutua, S., & Angulu, R. (2020). Breast cancer classification using local directional ternary patterns. *International Journal of Computer Applications, 176*(38), 14–21.

Ojala, T., & Pietikäinen, M. (n.d.). *Texture classification*. Machine Vision and Media Processing Unit.

Ojala, T., Pietikäinen, M., & Mäenpää, T. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 24*(7), 971–987. https://doi.org/10.1109/TPAMI.2002.1017623

Philomina, S., & Uma, V. (2020). Deep learning-based feature extraction for texture classification. *Procedia Computer Science, 171*, 1680–1687. https://doi.org/10.1016/j.procs.2020.04.180

Phuong, N. T., Ngoc, V. S., & Antaine, M. (2015). Statistical binary patterns for rotational invariant texture classification. *Neurocomputing*. https://doi.org/10.1016/j.neucom.2015.09.029

Ramola, A., Shakya, A. K., & Van, P. D. (2020). Study of statistical methods for texture analysis and their modern evaluations. *Engineering Reports, 2*(4), e12149. https://doi.org/10.1002/eng2.12149

Short WaveLab. (n.d.). *The Multimedia Signal Processing and Security Lab*. WaveLab. https://wavelab.at/sources/STex/

Taha, H. R., & Bee, E. K. (2014). Completed local ternary pattern for rotation invariant texture classification. *The Scientific World Journal*, Article 373254. https://doi.org/10.1155/2014/373254

Toulatzis, V., & Fudos, I. (2022). Deep tiling: Texture tile synthesis using a constant space deep learning approach. In *International Symposium on Visual Computing* (pp. 414–426). Springer. https://doi.org/10.1007/978-3-030-90439-5_33

Tuceryan, M., & Jain, A. K. (1998). Texture analysis. In C. H. Chen (Ed.), *The handbook of pattern recognition and computer vision* (2nd ed., pp. 207–248). World Scientific.

Xiaoyang, T., & Bill, T. (2010). Enhanced local texture feature sets for face recognition under difficult lighting conditions. *IEEE Transactions on Image Processing, 19*(6), 1635–1650.

Zhenhua, G., Lei, Z., & David, Z. (2010). A completed modeling of the local binary pattern operator for texture classification. *IEEE Transactions on Image Processing, 19*(6), 1657–1663. https://doi.org/10.1109/TIP.2010.2044957