American University in Cairo

AUC Knowledge Fountain

Theses and Dissertations

2-1-2016

A hybrid deep learning approach for texture analysis

Hussein Adly

Follow this and additional works at: https://fount.aucegypt.edu/etds

Recommended Citation

APA Citation

Adly, H. (2016).*A hybrid deep learning approach for texture analysis* [Master’s thesis, the American University in Cairo]. AUC Knowledge Fountain.
https://fount.aucegypt.edu/etds/359

MLA Citation

Adly, Hussein. *A hybrid deep learning approach for texture analysis*. 2016. American University in Cairo, Master's thesis. *AUC Knowledge Fountain*.
https://fount.aucegypt.edu/etds/359

This Thesis is brought to you for free and open access by AUC Knowledge Fountain. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of AUC Knowledge Fountain. For more information, please contact mark.muehlhaeusler@aucegypt.edu.

The American University in Cairo

School of Science and Engineering

A Hybrid Deep Learning Approach for Texture Analysis

A Thesis Submitted to

The Department of Computer Science and Engineering

In partial fulfillment of the requirements for the degree of Master of Science

By

Hussein Mohamed Adly

B.Sc. Computer Science

Under The Supervision of

Prof. Mohamed Moustafa

Fall 2016

Acknowledgements

I would first like to thank my thesis advisor Associate Professor Dr. Mohamed Moustafa at the American University in Cairo. Professor Moustafa’s office is always welcoming and provides assistance as required as I ran into a trouble spot or had a question about my research methodology or writing. He consistently allowed this research to progress at a steady pace due to his unconditional support and steering me in the right the direction whenever he thought I needed it.

Abstract

Texture classification is a problem that has various applications such as remote sensing and forest species recognition. Solutions tend to be custom fit to the dataset used but fails to generalize. The Convolutional Neural Network (CNN) in combination with Support Vector Machine (SVM) form a robust selection between powerful invariant feature extractor and accurate classifier. The fusion of classifiers shows the stability of classification among different datasets and slight improvement compared to state of the art methods. The classifiers are fused using confusion matrix after independent training of each using the same training set, then put to test. Statistical information about each classifier is fed to a confusion matrix that generates two confidence measures used in building two binary classifiers. The binary classifier is allowed to activate or deactivate a classifier during testing time based on a confidence measure obtained from the confusion matrix. The method obtained results approaching state of the art with a difference less than 1% in classification success rates. Moreover, the method was able to maintain this success rate among different datasets while other methods had failed to obtain similar stability. Two datasets had been used in this research Brodatz and Kylberg where the results came 98.17% and 99.70%. In comparison to conventional methods in the literature, it came as 98.9% and 99.64% respectively.

In conclusion of this research, a fusion of classifiers that exhibit uncorrelated errors such as Support Vector Machine (SVM) and Convolutional Neural Network (CNN) shall enhance the classification rates and also stability across different datasets and feature variations.

Table of Contents

List of Figures

List of Tables

CHAPTER 1. INTRODUCTION

1.1 Motivation

Texture analysis remains one of the challenging tasks in computer vision yet extremely useful for segmentation and classification with various practical applications. It is utilized in various problems, including remote sensing, forest species recognition, scene segmentation, content-based image retrieval, industrial inspection, etc. The textures work as clues in natural and artificial vision system to classify objects. With the advancement of machine learning algorithms, it is possibly feasible to achieve better accuracy than previously reported results due to depth regarding non-linearity and the sophistication of the new algorithms in data analysis.

In this study, we attempt to enhance the classification rates and stability across different datasets. The fusion of classifiers that has a degree of uncorrelated errors shall enhance the overall classification outcome. Using Convolutional Neural Networks (CNN) and Support Vector Machine (SVM) and fuse them with a binary classifier that works as on/off gate switch to set on the classifier with the highest confidence while keeping other classifiers off.

Convolutional Neural Network (CNN) is an upgrade from traditional neural networks to exploit features specific to image data. The CNN provides the deep learning process with extendable network architectures and various tools for non-linear learning and low overfitting. The CNN doesn’t require a handcrafted feature extractor and is able to automatically find in search space appropriate features to represent images within the training process without any human intervention. The CNN had shown great ability to classify thousands of images and millions of learning parameters and weights.

Support Vector Machine (SVM) is a general purpose classifier that had shown good results on various types of problems. The SVM allows various feature extractors to be used either singularly or with a fusion of more than a feature extractors. The SVM has one of the most accurate classifiers that relies on finding the best separation margin for vector data. It is computationally efficient compared to CNN.

1.2 Problem Definition

Texture classification and segmentation have proved themselves to be challenging problems in computer vision, yet still highly useful clues in the general image segmentation process. It's thus of vital importance to research, analyze and improve texture segmentation and classification algorithms by crafting better stabilized generalized processes and ensuring the applicability of usage with various types of texture problems without a significant loss in accuracy. The research fundamentally has two research questions.

1. Does the fusion improve the classification rates and stability?

   Conventionally different classifiers address different features. That produces highly specialized classifiers that fail to adapt to the changes of training data. The fusion attempts to provide a generalized solution that can adapt to various datasets with minimal parameters tuning. The fusion also shall improve the classification rates under the condition that both classifiers exhibit uncorrelated errors.

2. Can a confusion matrix statistical information support a binary classifier into delivering high classification rates?

   The confusion matrix provides powerful statistical information tools to measure the accuracy of a classifier with each class of data. The false positive indicates the percentage of wrongly classified data into a specific class. That information could be used to predict the likelihood that a classifier will wrongly classify any test sample into that class. The combination of true positive and the likelihood of misclassification for a specific class could contribute valuable information to a classifier to provide accurate predictions.

1.3 Objectives

The study concentrates on texture classification using statistical pattern extracted for SVM and automatically extracted patterns using CNN on two datasets namely Brodatz and Kylberg. Different classifiers have different learning and classification approaches and thus, a degree of uncorrelated errors. My aim of this research is to design a suitable architecture for Convolution Neural Network (CNN), finding the optimal set of features for Support Vector Machine (SVM) and fuse the classifiers to achieve better results that none of the experts could have reached independently.

1.4 Proposed Methodology

Using Convolution Neural Network (CNN) and Support Vector Machine (SVM) classifiers since both of these classifiers have produced state-of-the-art results in the literature [2]. The training and optimization of the CNN and SVM are done before fusing to determine the optimal classifier per class. It was shown that CNN could learn invariant features, but lacks behind with classification accuracy, while SVM is highly accurate with separation of feature vectors yet it suffers from significant variance in viewpoint, illumination and clutter [54].

1.5 Thesis Outline

The structure of the thesis is the following. Chapter 2 reviews other solutions in the literature with regard to their advantages and drawbacks. A comprehensive review of various feature extraction techniques and design aspects of convolution neural network. Chapter 3 introduce the design approach to convolutional neural network architecture and support vector machine learning parameters. Convolutional neural networks (CNN) [1]-[2] and support vector machines (SVM) [5]-[10] will be applied for texture classification. Chapter 4 shows the results of texture analysis experiments. In particular, different training and testing scenarios analyzed. Chapter 5 provides an comprehensive conclusion and future work.

CHAPTER 2. BACKGROUND

2.1 Basic notions of texture

The human visual system relies on a complex system of perception to visual cues that includes textures. The characteristics of these clues are not fully understood. Thus, many acceptable definitions of texture already exist. Texture analysis is the process of utilizing mathematically based models to describe spatial variations in images or scenes [29]. It is utilized in various fields including remote sensing, forest species recognition, scene segmentation, content-based image retrieval, industrial inspection, etc. Thus, texture classification and segmentation problems are remaining one of the fundamental pattern recognition tasks. The primary concern of texture analysis is with textural feature extraction and accurate matching of such features.

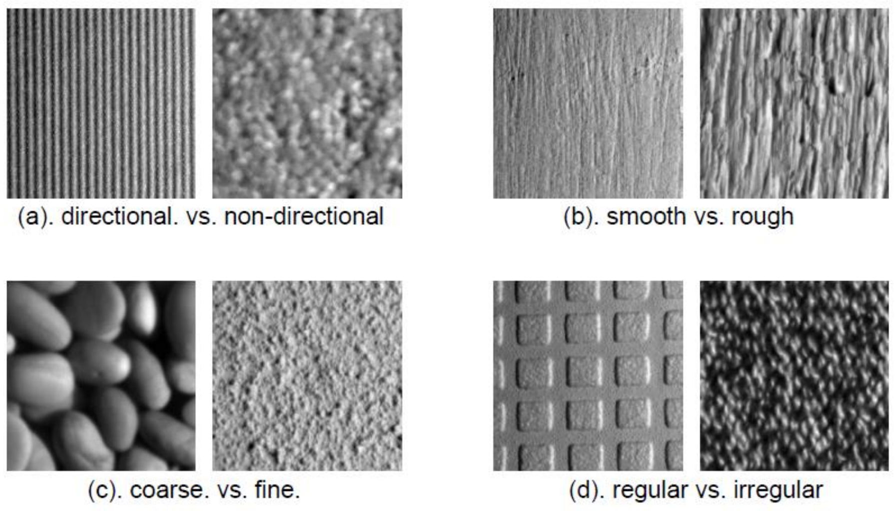

(a). directional. vs. non-directional
(b). smooth vs. rough
(c). coarse. vs. fine.
(d). regular vs. irregular

Figure 1 - Different types of textures by Optimising GEOBIA for seafloor classification (Vanessa Lucieer)

According to the definition, texture is a 2D characteristic, which demonstrates the slowly varying and almost periodic or regular structure of local intensity levels [4]. Figure 1 shows examples of different types of textures. An essential property of textures that they are repeating patterns that compose a scene or image. This general property is commonly used for extraction of local textural features, i.e. some statistical characteristics that could be utilized for an adequate description of texture patterns that shall be used for comparing and matching textures. Texture analysis is very useful when it is easier to describe an object through its texture, and not its edges. That is due to the nature of the object texture layout that makes using edges to describe and detect a class of objects like searching for a needle in a haystack. That is due to possible irregularities in texture orientation, scale, and sharp regular curvatures. A more general description of texture possibly using statistical measures is a more accurate representation of the object and easier to match against unseen images.

The two most common problems in texture analysis are classification and segmentation. Texture classification is about classifying textures into one of a previously determined set of classes, yet it could also include determining the classes themselves in unsupervised learning. The segmentation is the automatic identification of disjoint image regions and boundaries based on textural patterns.

2.2 Feature extraction methods

The classification process starts with feature extraction in which each feature is described in a numeric representation. These representations should take into account the possible transformations such as rotation and scale invariance. Feature extraction is the controlling factor the Support Vector Machine (SVM) performance, and it is done explicitly unlike CNN. Thus, there is a lot of attention and research focuses on the optimal feature extraction method for textures. The features could be separated into two groups, the local features, and global features. The researchers divided the feature space into two types. The local features in which its scope is very narrow and limited to a particular location without the need to worry about surroundings while the global features, on the contrary, takes the surroundings as a primary concern into account.

Lei et al.[31], developed a new feature extractor called the Complete LBP (CLBP). An image gray level local region is represented by its center pixel, and global thresholding is done before binary coding the center pixel. This method generates rotational invariant features, but the CLBP, on the other hand, suffers from poor robustness to the rotation.

A hybrid approach could be used to combine local and global features. Local Binary Pattern is used for local features, and various methods could fit perfectly for extracting global patterns. A fusion process follows the extraction of features before actual classification is performed.

Liao et al. [32] proposed Dominant Local Binary Patterns (DLBP), which is a linear process that is robust to noise. It works by calculating the frequency of occurrence of rotational invariant patterns. The extracted patterns are sorted in descending order from the most dominant to least dominant. To extract global features, Circularly Symmetric Gabor Filter is used, which is rotation invariant, and less sensitive to histogram equalization than DLBP. The aggregation of the local and global features into a classifier yields better results than either of them could achieve independently. The DLBP nonetheless suffers from high complexity regarding computational time.

Chung et al. [33], proposed Advanced Local Binary Patterns (ALBP), which is similar to LBP, but it captures spatial distribution patterns of information, which LBP misses. It relies on the histogram distribution of dominant patterns as features, then marking the location of these patterns and construct a binary representation of each image by storing the location of the pattern. ALBP uses Gray Level Aura Matrix (GLAM) measure to extract global spatial features. The GLAM can capture dominant local structure characteristics of texture, and global attributes with Aura Matrix measures that were rotation invariant and immune to histogram equalization. ALBP suffers from sensitivity to noise, and high computational time complexity.

Khellah [34] proposed Dominant Neighborhood Structure (DNS), which is a rotational invariant and sturdy to noise method. DNS works by utilizing a search window that measures local area intensity similarity using Euclidean distance. The distance is measured for a pixel about its surrounding pixels in a local area. An intensity similarity map is then produced to capture active neighborhood similarity. The map is robust to noise and could be used to generate new features. LBP is utilized for the extraction of local features before the global and local features are integrated. Dissimilarity measures are then taken by local and global features before they are summed to obtain an aggregate dissimilarity measure [30].

Jamie Melendez et al. [36] proposed a two-stage pixel-based classification method. It initially utilizes an unsupervised algorithm, and stack on the top of it a supervised one. A clustering algorithm cluster patterns into similar groups with the aim of high accuracy in the grouping, and accurate classification of image regions. Using the output of the unsupervised algorithm, patterns from each class determined by the unsupervised algorithm is used to train the supervised algorithm. The features are extracted using Gabor filter bank with six orientations, and four scales. A Large window size is used to classify similar texture regions, while smaller window size is used to classify boundary pixels. This method is used to avoid noise during classification and unnecessary computations by using variable window sizes for pixels. The pixel's classification is done using One-Against-One SVM.

A. Rampun et al. [40] proposed a method based on Grey-Level Co-occurrence Matrices (GLCM), which uses various feature measures such as contrast and entropy. For each of the extracted features, normalization of values to ranges between 0 and one is performed. Smoothing and noise reduction are performed using Discrete Cosine Transform (DCT) with 5x5 filter size and dimensionality reduction to remove irrelevant or redundant features using Principal Component Analysis (PCA). A GMM, which is the weighted sum of Gaussian components, is then used to cluster the data, and hole-filling is applied to cluster any islands.

B. M. Carvalho et al. [28] proposed Fuzzy Segmentation, which is a region semi-automatic growing segmentation method. For a given pixel and object, a 0 or 1 is assigned to the pixel based on whether belonging to the object or not. This process works by utilizing connectedness score to every pair of connected pixels, which allows forming chains of pixels belonging to the same object. It works by using a neighborhood filter of size 3x3 and keeps going along connected region as long as every two consecutive pixels have differences in statistics below a priory-defined threshold. The reported results signify an improvement over other approaches, yet the reported results lack any quantitative information about the error rate of experiments.

M. Mehri et al. [27] proposed a method to segment book's page's text and graphics by randomly selecting foreground pixels and using them in estimating the number of similar clusters existing in a region using Consensus Clustering method (CC). The CC clustering algorithm works by averaging aggregated results of various clustering algorithms such as Agglomerative Nesting, Divisive Analysis Clustering, Hierarchical Ascendant Classification and K-means. The entire page features are then calculated, and an unsupervised algorithm is used for clustering features into homogeneous regions taking into consideration the number of clusters estimated from the pixel samples. The features are computed using co-occurrence, Gabor, and co-relation methods on gray level images using various sliding windows.

2.3 Machine learning methods for Texture Classification

2.3.1 *Support Vector Machines (SVM)*

One of the most widely used classifiers for texture classification is SVM. The choice of CNN and SVM is shown to dominate reported results in the literature [2]. Originally, SVM was intended for only two-class problems as shown in Figure 2.

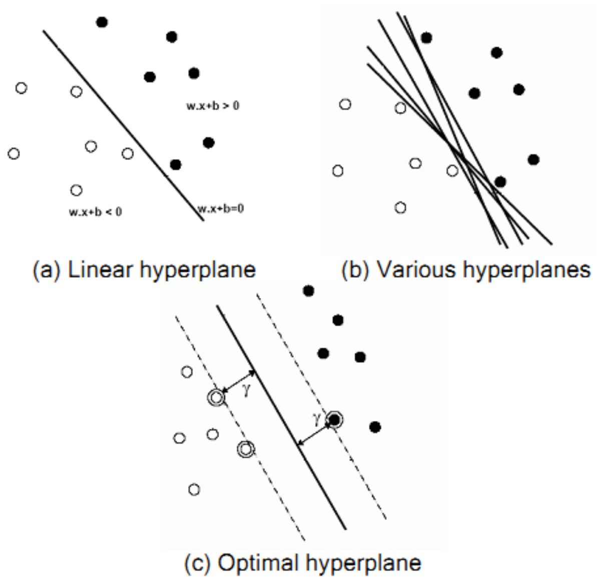


Figure 2 - Wavelets and Support Vector Machines by Kashif Mahmood Rajpoot & Nasir Mahmood Rajpoot

The SVM classifier strength is due to being neither relatively computationally intensive nor prone to noisy data [9] compared to neural networks. It performs well with higher dimensional or attributes spaces while high dimensionality is a challenging problem for many classifiers. As a result, SVM is not highly prone to overfitting since generalization in SVM does not depend on dimensionality. SVM also neither fall for local optimum [38] nor critically suffer from small training data. It works by minimizing an estimate of test error rather than the training error which is also called structural risk minimization. In order by taking advantage of the superiority of SVM, problems of non-linear nature are transformed into high dimensional linear problems.

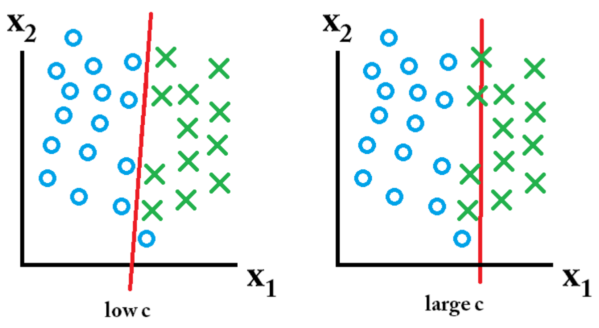


Figure 3 - Effect of C on hyperplane

The SVM maximizes the margin in hyperplane to reach maximal separation as shown in Figure 2(c). The SVM depends on two vital parameters, the Gamma, and C to achieve optimal margin. Gamma defines the influence of a support vector on the classification of other support vectors with regard to the distance between the two. Large values of Gamma implies that vectors far in the distance have no effect over each other classification and vice versa. Colossal values of Gamma will make the only affecting factor the support vector it’self, and overfitting will be inevitable regardless of C. Too small values of gamma will lead to over influence of the whole training set on the classification decision of any vector, and thus it’s unlikely that a generalization would be reached. The C defines how tolerable is permitting errors to happen, as occasionally that could lead to a better convergence in case of the substantial existence of outliers. The higher the C, the severer the plenty of misclassification. It may not usually be considered a good practice to assign substantial plenty on misclassification since the spreading of the data might be very far from linear. Loose values of C might lead to very crude situations of model overfitting as shown in Figure 3.

SVM takes features as an input and finds a margin to separate these features. To analyze texture features, some statistical characteristics to represent the textures shall be introduced. The determination and computation of these features are called features extraction. The process of features extraction is an essential intermediary operation required by SVM and CNN classifiers as shown in Figure 4 as it extracts descriptors that can describe the textures mathematically. Optimal features choices will determine the likelihood of success at later stages of classification.

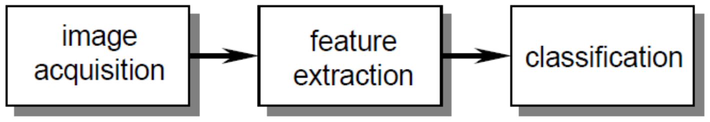


Figure 4 - Classification Process

Fundamental characteristics shall be present in any optimal texture analysis methodology. The method should exhibit invariance in illumination, scale, rotation and projection invariance. It also should be robust against noise, has low computational complexity, and variable windows or patch size if applicable to best suit the problem [29].

Texture features are divided into three categories statistical, structural / geometrical and digital signal processing methods. The statistical method relies on the statistics of the spatial distribution of usually constant or slowly varying gray level values. The spatial statistics are used as a pattern that could be used to match to similar statistical pattern information. Methods such as Co-occurrence which is the probability of occurrence of two events together and autocorrelation, which captures statistical information about co-occurrence such as frequency in per region. Both show the repetition of textures as well as local intensity variations [29]. They are functions of a statistical nature and belong to the statistical methods arena. Geometrical methods, which decompose textures into simple geometrical primitives in addition to their placement by using techniques such as edge detection with a Laplacian-of-Gaussian. Statistical methods could be used to describe patterns in the information generated by a geometrical method. Signal processing methods analyze the frequency domain of spatial information. Fourier analysis is used to analyze frequency domain optimally when textures exhibit high periodicity. Although realistic natural patterns do not usually have a high recurrence in addition to variations in scale, spatial orientation, and illumination that makes it difficult to describe textures using frequency domain methods [3].

2.3.1.1 Co-occurrence Matrix (CM) Analysis

A co-occurrence matrix (CM) is the co-occurrence distribution defined over a spatial domain concerning the distribution of co-occurring values. The co-occurrence matrix belongs to the statistical approach methods, and it measures the distance and angular spatial relationship of an image sub-region of a particular size. The CM could be used to measure features of contrast, which is a measurement of the local variation of pixel intensity that is the difference between the highest and lowest pixel values in a region. Other features as correlation which is the joint probability of co-occurrence of a particular set of pixels, homogeneity which is how similar is the distribution of pixel around GLCM diagonal, angular second-moment measures the consistency of textural information which is how similar or homogeneous a region is in terms of pixel values and dissimilarity which is the variance of the gray level allocation of pixels [35][26].

Rotational invariance GLCM features should be averaged over all directions. Although it has somewhat achieved its purpose, in the process, the directionality information is lost.

Given an image I, of size $N \times N$, the co-occurrence matrix could be defined by (Eq. 1). In (Eq. 1 offset $(\Delta x \,, \Delta y)$, is defining the expected distance between a specific pixel and a neighboring one.

$$P(i,j) = \sum_{x=1}^{N} \sum_{y=1}^{N} \begin{cases} 1, & if\ I(x,y) = i\ and\ I(x + \Delta_x, y + \Delta_y) = j \\ 0, & otherwise. \end{cases}$$

(Eq. 1)

The co-occurrence matrix suffers from the absence of rotation invariance, such that if a new offset isn't located at least 180 degrees away from the old offset; a new occurrence different matrix will be generated for the same picture. This could be mitigated by producing multiple co-occurrence matrices such that each of them has a different rotational angle that doesn't exceed 180 degrees while preserving the distance between neighboring pixels parameter Δ to achieve approximate rotational invariance, four GLCM matrices are to be created at angles $0°$, $45°$, $90°$, $135°$.

The initial offset setting: $0°$ and $\Delta = 1$ indicated that the pixel-of-interest is compared to neighboring pixel at distance 1 pixel to the right.

The Haralick features are measured at distances Δ of 1, 3 and 5. Figure 5 shows various CMs constructed at different rotation and distance measures.

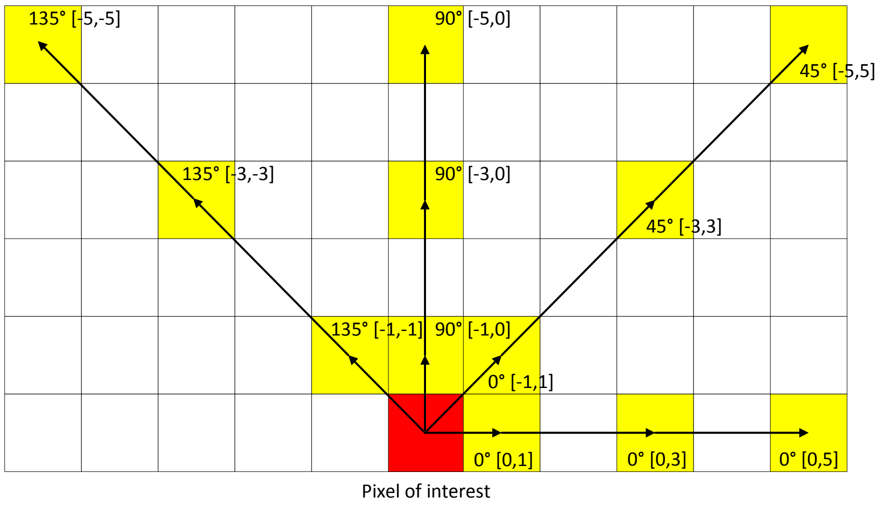


Figure 5 - Pixels of interest for CM construction

Haralick features [11], also known as Gray Level Co-occurrence Matrix (GLCM) features, are one of the most widely used texture features. All of the Haralick features are calculated based on the GLCM of the input image. GLCM detects textures in images based on reoccurrence of a pattern of pixel values at a certain spatial distance as shown in Figure 6**Error! Reference source not found.**.

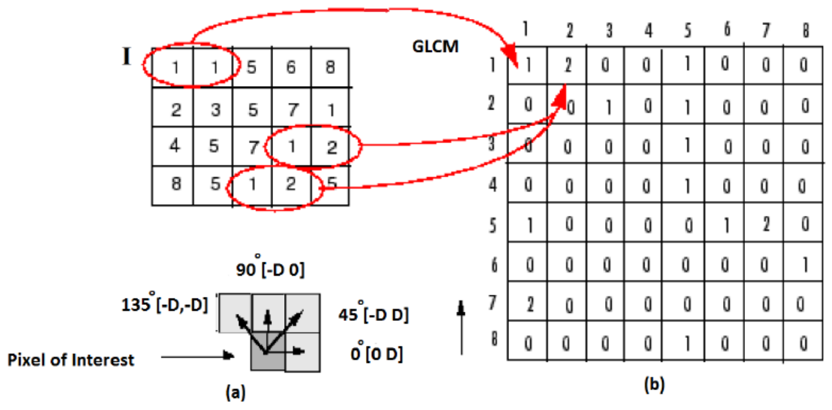


Figure 6 - Gray Level Co-occurrence Matrix for neighboring pixels with a distance of 1 by Muhammad Salman Haleem et al.

As a result, the Gray-level co-occurrence matrix shown in (Eq. 2).

$$\mathbf{G} = \begin{bmatrix} p(1,1) & p(1,2) & \cdots & p(1,N_g) \\ p(2,1) & p(2,2) & \cdots & p(2,N_g) \\ \vdots & \vdots & \ddots & \vdots \\ p(N_g,1) & p(N_g,2) & \cdots & p(N_g,N_g) \end{bmatrix}$$

(Eq. 2)

It is used as input for Haralick features calculation.

In Haralick features, functions are applied at different spatial angular positions, which are 0°, 45°, 90°and 135° respectively. The average of the generated features of four angles at specific distances *Δ* of either 1, 3, or 5 is computed to generate a new feature that is relatively rotation invariant texture representation [4]. Some of the Haralick features such as contrast, inverse difference moment and entropy were found to be very similar to human vision perception [3]. To make the initial analysis, we have chosen few texture samples from the Brodatz texture dataset as shown in Figure 7.

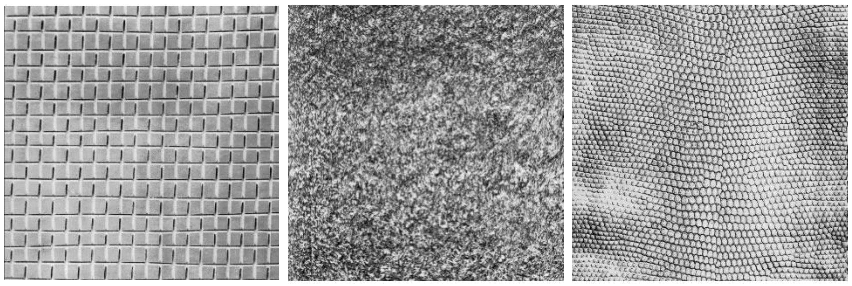


Figure 7 – Samples from Brodatz dataset

An example of calculating Haralick features is given in Figure 8

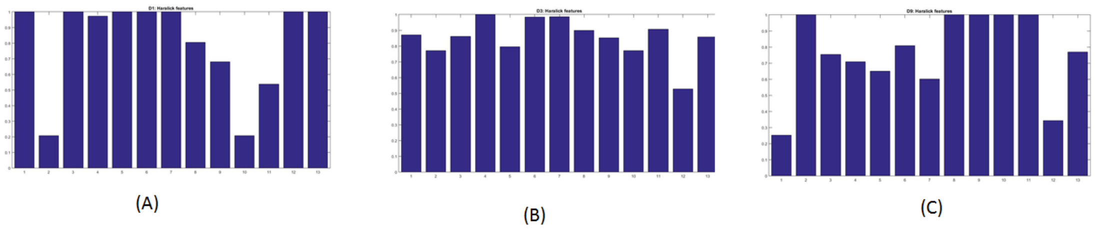
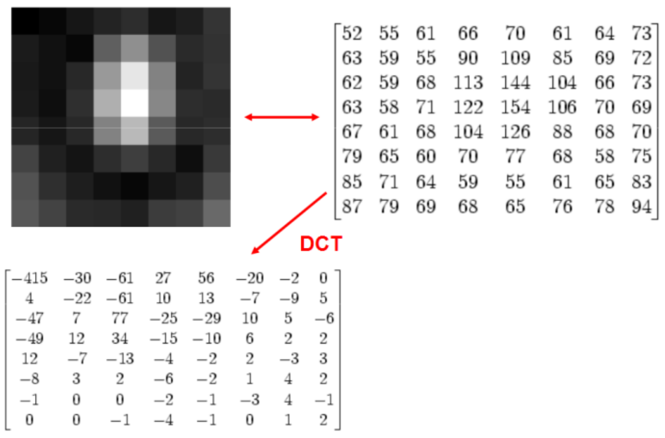


Figure 8 – Haralick features (a – sample 1, b – sample 2, c – sample 3).

One can see that the difference exists which facilitates discrimination between different textures.

2.3.1.2 Discrete Cosine Transform (DCT) features

Discrete Cosine Transform (DCT) maps an image from the spatial domain to frequency domain [14]-[16]. Typically, the DCT transform is applied to 8*8 image blocks as shown in Figure 9 to factorize the coefficients. Figure 10 illustrates the basis functions which act as a library of all possible cosine waves used to generate any image in 8*8 dimensions. Different weights are assigned to each pattern in the basis function to decompose the picture into its constituents.



$$\begin{bmatrix} 52 & 55 & 61 & 66 & 70 & 61 & 64 & 73 \\ 63 & 59 & 55 & 90 & 109 & 85 & 69 & 72 \\ 62 & 59 & 68 & 113 & 144 & 104 & 66 & 73 \\ 63 & 58 & 71 & 122 & 154 & 106 & 70 & 69 \\ 67 & 61 & 68 & 104 & 126 & 88 & 68 & 70 \\ 79 & 65 & 60 & 70 & 77 & 68 & 58 & 75 \\ 85 & 71 & 64 & 59 & 55 & 61 & 65 & 83 \\ 87 & 79 & 69 & 68 & 65 & 76 & 78 & 94 \end{bmatrix}$$

$$\begin{bmatrix} -415 & -30 & -61 & 27 & 56 & -20 & -2 & 0 \\ 4 & -22 & -61 & 10 & 13 & -7 & -9 & 5 \\ -47 & 7 & 77 & -25 & -29 & 10 & 5 & -6 \\ -49 & 12 & 34 & -15 & -10 & 6 & 2 & 2 \\ 12 & -7 & -13 & -4 & -2 & 2 & -3 & 3 \\ -8 & 3 & 2 & -6 & -2 & 1 & 4 & 2 \\ -1 & 0 & 0 & -2 & -1 & -3 & 4 & -1 \\ 0 & 0 & -1 & -4 & -1 & 0 & 1 & 2 \end{bmatrix}$$

Figure 9 – Extract 8 x 8 block and apply DCT transformation by Nuno Vasconcelos

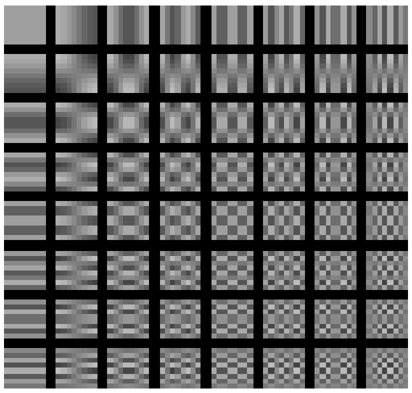


Figure 10 – DCT basis functions, Enhancement of Low Contrast by A. K. Bhandari, A. Kumar, and P. K. Padhy

The spatial frequency is increasing from top-left to the bottom-right. The high frequencies correspond to the minute changes in images while the low frequencies correspond to some large-scale intensity variations. The DCT transform works as follows on input images.

$$F(u,v) = \frac{1}{\sqrt{2N}} c(u)c(v) \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} f(i,j) \times \cos\left[\frac{(2i+1)u\pi}{2N}\right] \cdot \cos\left[\frac{(2j+1)v\pi}{2N}\right]$$

for $u,v = 0,1,\ldots,N-1$

where

$$c(x) = \begin{cases} \frac{1}{\sqrt{2}} & \text{for x=0} \\ 1 & \text{otherwise} \end{cases}$$

(Eq. 3)

There are several approaches for utilization of DCT features. We have found that these are two measures are of interest

- Usage of energy function which calculates the differences and changes between the values of DCT cells, which could be used for the texture segmentation
- Usage of statistical moments, which is the analysis of DCT coefficients histograms of the particular DCT cell distribution.

To analyze the potential of DCT features, let us consider few texture samples as shown in Figure 11. The image is decomposed into 8*8 blocks that are the original image we have 64 blocks horizontal axis and 64 blocks to the vertical axis of 8*8 blocks since 512/8 is 64 on both axis, where highest frequency is located at the top-left corner and the lowest is located at the bottom right of each DCT block. A histogram is, plots for each frequency level across all blocks. Since DCT splits the image into 8*8 blocks the total number of histograms drawn is 64 histograms for each 512 x 512 Brodatz image.

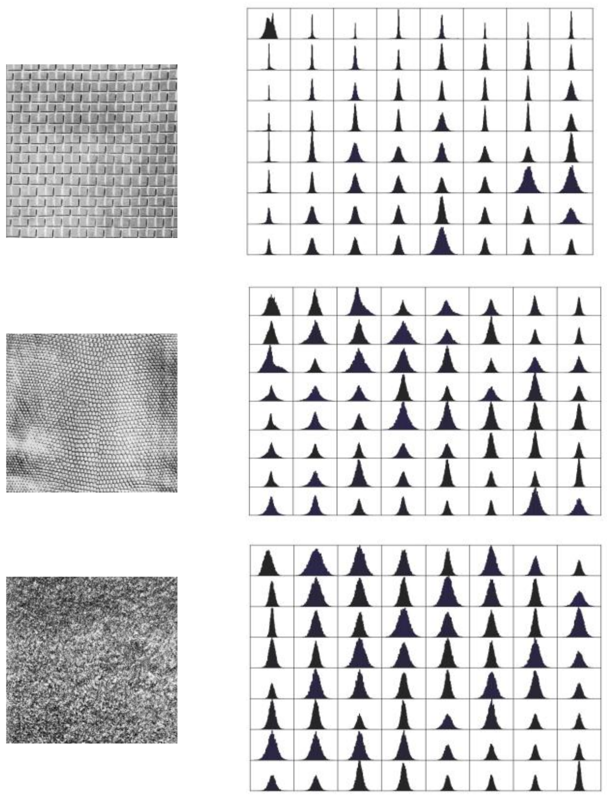


Figure 11 – contains a set of calculated histograms for DCT coefficients distributions within 8x8 DCT blocks.

To transform the image data into features the statistical moments which are Mean, Variance, Skewness, Kurtosis, and Standard Deviation of DCT coefficients shall be calculated.

Where Kurtosis is defined as

$$K = n \frac{\sum_{i=1}^{n}(X_i - X_{avg})^4}{(\sum_{i=1}^{n}(X_i - X_{avg})^2)^2}$$

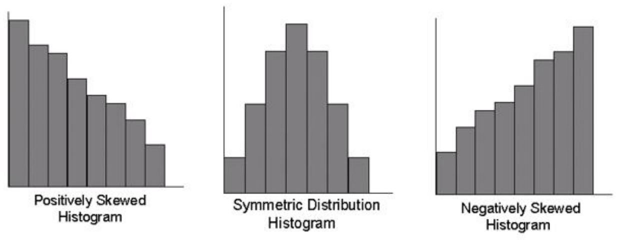
(Eq. 4)

And Skewness is defined as

$$S = \sqrt{n}\,\frac{\sum_{i=1}^{n}(X_i - X_{avg})^3}{(\sum_{i=1}^{n}(X_i - X_{avg})^2\,)^{3/2}}$$

(Eq. 5)

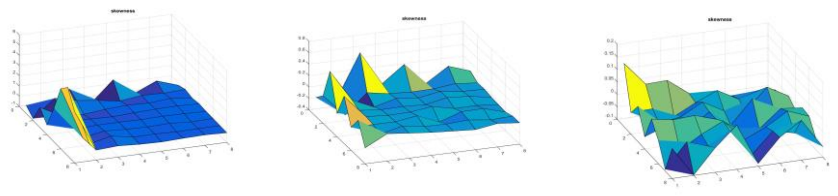
Skewness measures the asymmetry of data. A zero skewness indicates that the histogram is symmetric. Meanwhile, positive skewness suggests that the mean is greater than the median, and vice versa as shown in Figure 12.



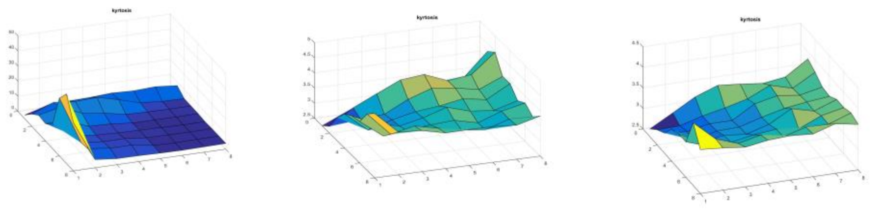
Source: http://library.thinkquest.org/10030/3smodsas.htm

Figure 12 – Analysis of DCT coefficients distributions

Kurtosis measures peaking of the histogram. In other words, measuring the peakedness or flatness in comparison with the normal distribution. Platykurtic is a negative Kurtosis, where the curve is flat, and Leptokurtic is positive Kurtosis in which the curve is wide, and peaked. Figure 13 illustrates examples of calculating skewness, kurtosis and distribution width



(a)



(b)

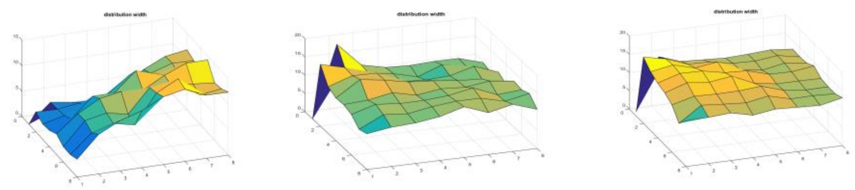


Figure 13 – Analysis of DCT coefficients distributions (a – skews, b – kurtosis, c – distribution width)

One can observe that the histogram distribution of shape characteristics differs for texture samples. Thus, it is possible to use such features as input for support vector machine.

2.3.1.3 Miscellaneous Methods

The Local Binary Pattern (LBP) [12] captures texture primitives and local spatial information. It works by calculating a value that reflects the relationship between a 3 × 3 neighborhood through applying a threshold to the pixels of the region. Additionally, assigning weights to the neighboring pixels using binomial weights is an option to tweak the LBP. The 3x3 block could generate 256 different permutations of texture descriptors since 2^8 possible combinations are present. A histogram could be produced given the pattern of 0's, and 1's that describes image regions as shown in Figure 14. In LBP, the value of the center pixel depends on the neighborhood; it is one for a given center pixel if the count of neighboring pixels higher in value is more than those lower in value and zero otherwise. The reliability of LBP patterns are not equal; some patterns are robust to geometric transformations while others are not. Usually, two main problems are present within the aforementioned method. It relies on a window of a particular static size that should be defined a priori while on the other hand, the optimal window size is problem dependent and when an object accidently lies on the boundary of the window, that shall make it unlikely that the produced representation will facilitate pattern detection in that area [3].

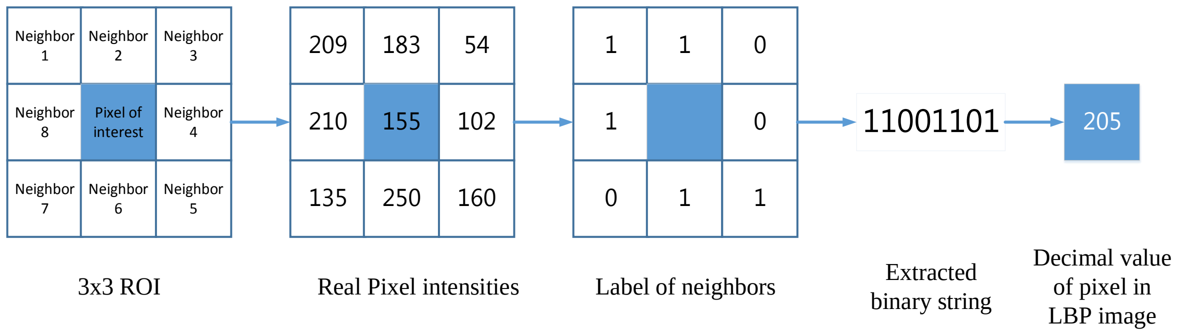


Figure 14 – Local binary pattern

Other features commonly used for the texture analysis are Gabor features that rely on the frequency domain bank of filters such as high pass and low pass [13]. Typically, the initial texture image is convolved with a bank of Gabor filters for different spatial frequencies and orientations. The pixels of obtained filtered versions are used as features.

Analysis of transform domain of the texture images also can give additional benefits. The Fourier spectrum magnitude can be used for features calculation.

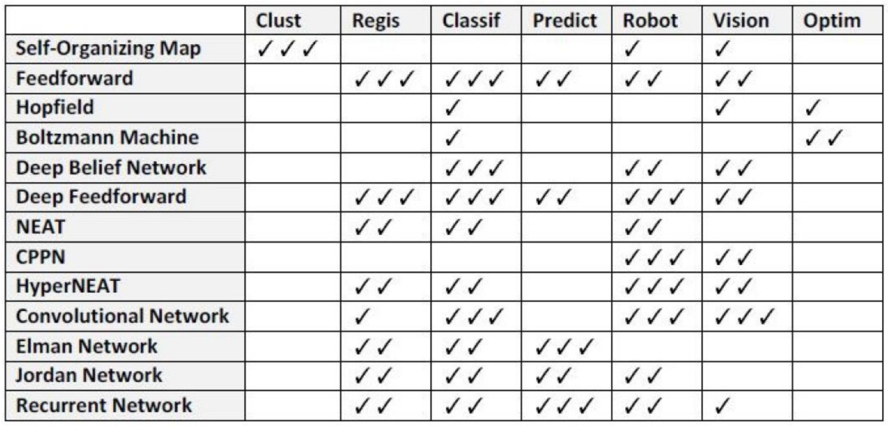
Another feature group is related to spectral histograms [17]-[19]. A spectral histogram of an image is the marginal distribution vector of filter responses and integration of different filters responses. Various filters from the constructed filter bank are convolved with the input image. As a result, the regions homogeneity is adequately analyzed which allows using the spectral histogram for efficient texture analysis.

The potential of analysis of textural images with different resolution levels is demonstrated in [20]. It is also possible to utilize the histogram features in the bimage version blurred with different Gaussian kernels.

The Gray Level Run Length Statistics measure the likelihood that consecutive points in the picture have same gray level while the gray level histogram used to measure the distribution of texture intensity. Laws Texture Energy Measure (TEM) uses convolution 5x5 mask to extract features and Fourier Power Spectrum (FPS), which is an upgrade to Fourier transform, works on wavelike patterns [37].

2.3.2 Convolutional Neural Networks (CNN)

The CNN is the classifier of choice among the many variations of neural network based classifiers for applications like computer vision, and classification [5]. CNN has many features that make it the optimal choice for computer vision and image classification tasks as shown in Figure 15 in the comparison between different neural networks (NN) based classifiers.



	Clust	Regis	Classif	Predict	Robot	Vision	Optim
Self-Organizing Map	✓✓✓				✓	✓	
Feedforward		✓✓✓	✓✓✓	✓✓	✓✓	✓✓	
Hopfield			✓			✓	✓
Boltzmann Machine			✓				✓✓
Deep Belief Network			✓✓✓		✓✓	✓✓	
Deep Feedforward		✓✓✓	✓✓✓	✓✓	✓✓✓	✓✓	
NEAT		✓✓	✓✓		✓✓		
CPPN					✓✓✓	✓✓	
HyperNEAT		✓✓	✓✓		✓✓✓	✓✓	
Convolutional Network		✓	✓✓✓		✓✓✓	✓✓✓	
Elman Network		✓✓	✓✓	✓✓✓			
Jordan Network		✓✓	✓✓	✓✓	✓✓		
Recurrent Network		✓✓	✓✓	✓✓✓	✓✓	✓	

Figure 15 – Artificial Intelligence for Humans by J. Heaton, A Comparison of Neural Network-based Learning Algorithms.

The convolutional neural network (CNN) is very similar to the traditional neural network. It consists of consecutive layers of neurons with learnable weights and biases. The primary difference between NN and CNN is the concept of perceptive field. -Unlike usual NN, which transforms the input data through a set of hidden layers that are fully connected and neuron in the same layer are independent of each other, the CNN exploits a more sensible way to share weights among neurons and limit connectivity to desired level depending on the problem. CNN is also well suited for GPU computation for both training and testing procedures.

2.3.2.1 Convolutional layer

The motivation behind the layered approach was the mammal's visual system. Convolutional neural networks are comparable to the human optical system as shown by researchers, especially in terms of the inner workings of object recognition [41]. Convolutional neural networks have demonstrated success empirically and biologically sound, yet the theoretical basis for convolutional neural networks success is not yet well understood scientifically.

The CNN has three layers called 3D layers, namely height, width, and depth. It could also be visualized as a 1D stack of 2D filters. A subset of a layer's neurons works as receptive fields for the layer next to it as shown in Figure 16.

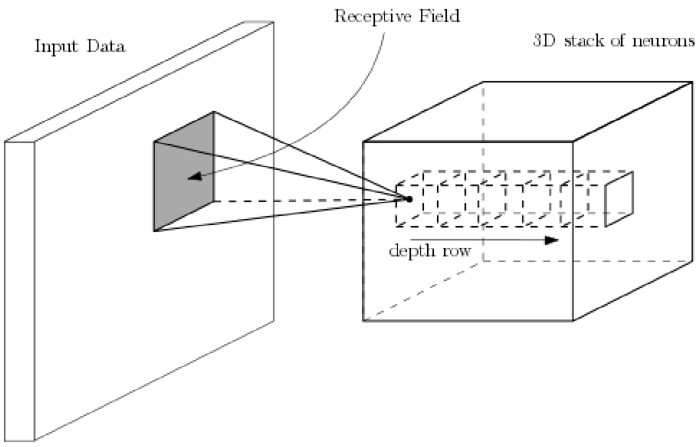


Figure 16 – Receptive Field

For example, if an image is in RGB format, three stacks of neurons are needed to accurately describe each of the layers R, G, and B.

The process CNN applies to its input could be described as follows.

$$y_{jk}^{i} = \sum_{r=1}^{F_j} \sum_{l=1}^{F_k} x_{(j+r-1)(k+l-1)}^{i-1} w_{rl}^{i} + b^{i}$$

(Eq. 6)

The y_{jk}^{i} represents the output pixel at (j,k), F_j accounts for the number of rows and F_k represents the number of columns of the 2D filter, w_{rl}^{i} accounts for the value of the filter kernel at the location (r,l), $x_{(j+r-1)(k+l-1)}^{i-1}$ accounts for the input value to the layer at $(j+r-1, k+l-1)$ and b^{i} is the bias.

Usually, with images, it was empirically shown that connecting neurons to only their corresponding neurons in the previous layer to be of great benefit. In CNN, neurons are only connected to only a subset of the neurons of the previous layer, which is called receptive field. The chosen subset of neurons of the receptive field isn't of constant dimensions or size, but is determined by the scale of the filter selected. The width, height, and depth of filters are treated in a different manner. For the height and depth, connections are local, while going to the full depth of receptive fields.

2.3.2.2 Rectified Linear Unit

ReLU is the defacto standard nonlinearity tool for convolutional neural networks. It brings more than zero or one to the layer next to it. Thus, preserving valuable information while still maintaining low complexity in computation by using (Eq. 7).

$$f(x) = \max(0, x)$$

(Eq. 7)

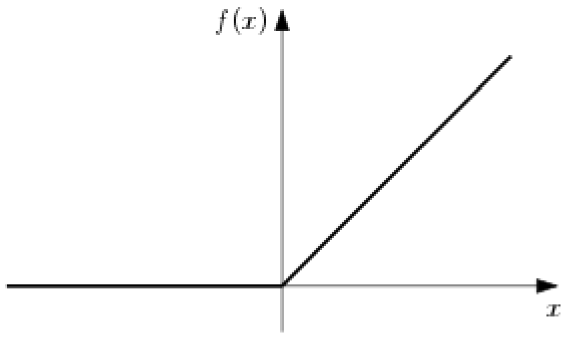


Figure 17 - Activation function of ReLU layer

The graph in Figure 17 illustrates the ranges of possible outputs to the ReLU. The ReLU is preferred due to its empirical reliability in accelerating the gradient descent and adding non-linearly without the heavy computational overhead. It is sufficient to determine the activation is to threshold the matrix with zero.

The ReLU may on the other hand cause network to die out. The ReLU may cause large parts of the network to be deactivated permanently through weight adjustment. The learning rate contributes to that problem, especially with the extremely high learning rate. The problem could be solved by selecting an optimal learning rate, which shouldn't be very high. Another version of ReLUs was introduced to address the problem called leaky ReLUs that let a small a positive gradient to help in recovery for negative inputs.

2.3.2.3 Pooling layer

Pooling is used to replace a spatially connected group of neurons into one by either using averaging or choosing maximal value neuron in the group. Pooling layers are used for dimensionality reduction to reduce the number of computations through lessening the amount of data moving throughout the network. Pooling layers also make CNN better at translation invariance.

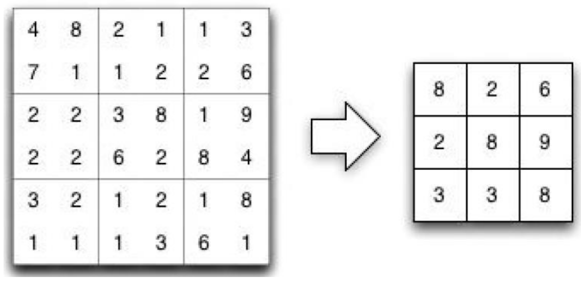


Figure 18 - Artificial Intelligence for Humans by J. Heaton, Grid downsampling using the window of 2x2 to 1.

The pooling operation works on depth levels of all filters and performs a non-linear downsampling operation that only affects the spatial size without affecting the amount of depth. The most popular operations are 'max' as shown in Figure 18 or 'average' operations as shown in Figure 19. The max pooling was shown to be superior to other methods [51]. The pooling layer requires two input parameters, the spatial window size used for downsampling and stride. Due to downsampling the spatial dimension of each layer in a volume of filters is reduced and could be described using the following equations $W_1 \times H_1 \times D_1$ where, $W_1 = (W - F)/S + 1$, $H_1 = (H - F)/S + 1$ and $D_1 = D$ Where W is the width, H is the height and D are the depth.

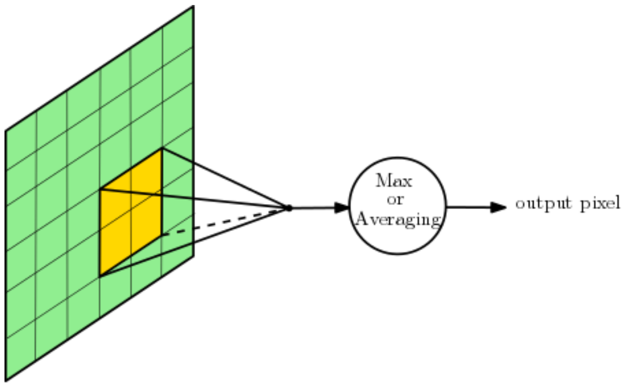


Figure 19 - Pooling layer action

2.3.2.4 Fully connected layer

A fully connected layer is a layer that has connections to every output neuron of the preceding layer, which is similar to traditional neural networks. A fully connected layer is usually used as the output layer for CNN with output neurons corresponding to the number of classes. The fully connected layer is a variant of the convolution layer in which the number of connections and weights is higher than convolutional layers as connections are made to all the output neurons of the previous layer for every neuron.

2.3.2.5 Softmax and Softmaxloss layers

The softmax layer is actually the classification layer where the decision is made regarding input class prediction at the final stage of the CNN architecture. To determine the negative impact of the likelihood of a class SoftMaxLoss classifier at (Eq. 8) is used. The F_i represents the element i in the computed output scores vector. In this equation exponentiation of F_i gives un-normalized probabilities while and division provides normalization. The negative of log computes the negative log likelihood of class that shall be minimized. It utilizes cross-entropy loss function to distinguish between various predictions.

$$L_i = -\log\left(\frac{e^{f_i}}{\sum_j e^{f_j}}\right)$$

(Eq. 8)

2.3.2.6 Dropout layer

A traditional problem of neural networks and convolutional neural networks is overfitting in which the neurons learn noise and adapt their weights accordingly. Thus becoming useless at detecting useful features and classification, which decreases the CNN performance.

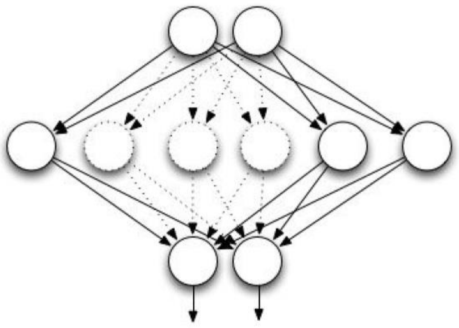


Figure 20 - Neural Network neurons deactivated at hidden layer By J. Heaton

A dropout layer is used to solve the problem and preventing CNN from overfitting. The dropping works randomly by dropping neurons during the training, and cutting these neuron connections as shown in Figure 20. It has been demonstrated that this method is superior to other regularization methods and efficiently solves the overfitting phenomena [52]. The dropout usage in various supervised learning tasks showed that it works for different problems such as computer vision, speech recognition, document classification and computational biology.

2.4 Experts Fusion

Studies have shown that a classification method could be superior regarding accuracy with a problem or a subset of them, but not with a whole domain of problems and across different datasets. Thus, integration of various methods could be the solution by utilizing different learning models implemented by each classifier in such a way that it improves the aggregated result.

The integration process starts by defining the data distribution methodology among experts, in which each of them gets a subset of the data either stochastically or explicitly. The stochastic allocation of data might bring some problem as it relies on initial weights that may not be deterministically efficient and could lead to complex scenarios as the zero-coefficient problem since the initial weights are randomly selected leading to difficulty in convergence. On the other hand, the explicit distribution of the data facilitates the accurate determination expert's assignment by using a specialist per existing class or cluster of classes so that each expert is specialized in only one cluster. The allocation should take into consideration the expert strengths and allocated the task to it that should match it.

2.5 Available datasets

There are three types of texture datasets available medical imaging, natural texture and texture of materials.

The medical imaging is very rich since a significant amount of research has been done in this field. MRI brain texture set is made from 28 patients suffering from WM encephalopathy or Alzheimer disease. Digital Database for Screening Mammography (DDSM) is a set made from 2620 patients with breast tumor. Information like patient age, stage of the tumor and ratings for abnormalities are stored with each image of the breasts. These pictures dimensions are 3000x4500 pixels.

Natural images datasets are dominant regarding a number of classes per dataset. The widely used Brodatz dataset is composed of pure natural textures. It is commonly used in computer vision and signal processing community. It has 112 classes with image dimensions of 512x512 pixels. Many researchers opted for not using the whole dataset of textures for simplicity, which is not the case at our research. The real reason behind the superiority of Brodatz is that the images were taken under controlled lighting conditions which resulted in very high-quality images. The images are almost clear from noise or non-textural components. Some of the texture classes look very similar that a human may not be able to distinguish between them correctly. Vision Texture dataset (VisTex) is a well-known dataset produced by MIT Media Lab containing 167 colored textures with two sizes, either 786x512 pixels or 512x786 pixels. Unlike Brodatz, the images were not taken in controlled environment lighting. Thus images are natural, yet they contain a considerable amount of noise such as shadows. Texture library is a library composed of textures from 17 different albums that has high-resolution textures. Even though the library exhibits significant variability, there were not any published results found for this dataset.

The texture of materials has real life textures which include natural textures and scenes. Measurement of Texture (MeasTex) is a dataset that has both natural and artificial textures. It has 4 class types of textures asphalt, concrete, grass and rock. Each image dimensions are 512x512 pixels. Photometric image set with a wider variety of illumination directions (PhoTex) is a dataset that contains images of surfaces in which each surface is pictured from 40 different viewpoints to produce different illumination. Although some rotation is present in the data, its primary focus isn't rotation but illumination. The images are taken in a controlled environment which allows through calibration the calculation of noise across different viewpoints of a picture. The image's dimensions are 1280x1024 pixels. Amsterdam Library of Textures (ALOT) contains 250 texture classes where each class has 100 images. Images have systematically varying the viewing angle and illumination angle. In total, it has eight illuminations in three orientations and four viewpoints. University of Maryland (UMD) dataset contains 1000 images taken using a family camera in uncontrolled conditions. It features 25 classes with a resolution of 1280 x 900 pixels. The dataset has images from different viewpoints and scale. OUTex dataset of surfaces and natural scenes. It has a significant variation in the illumination direction and rotation. Columbia-Utrecht Reflectance and Texture database (CUReT) is a tile dataset that has 61 different materials. Each image resolution is 512 x 512 pixels. The images in the dataset are taken in an uncontrolled environment with various noise types and illumination variance. The dataset suffers from the static scale and poor rotation. The textile dataset had eight textures represented by gray level images of dimensions 256 x 256 pixels and was captured in a controlled environment. UIUC dataset has 25 classes; each class is represented by 40 images. The images are grayscale with a resolution of 640 x 480 pixels. The images have different viewpoints and scales while the illumination is uncontrolled.

Textures under varying Illumination, Pose and Scale (KTH-TIPS) dataset is an extension of CUReT dataset by including images of various scales to ten of the CUReT textures. It has 81 images per class, where each image dimensions are 200 x 200 pixels with few exceptions of size. The pictures are taken at nine distances from the camera to provide various scales. In addition to scale variance, images are captured from three directions of illumination in three poses. Grain mixtures dataset has 11 different combinations of rice and barley grain. Each image resolution is 128 x 128 pixels [42].

CHAPTER 3. APPROACH

This thesis primary focus is finding reasonably optimal architecture for CNN, and best available feature extraction methodology or ensemble of mythologies for SVM before SVM parameter optimization and finally fusion of both experts in a form that would lead to better utilization of both experts' strengths. The data is split into four sections between test, validation, and train. Each classifier is trained with the same data, then put to test. The test statistical information is fed to confusion networks to guide building of a binary classifier that is allowed to choose which classifier shall perform the classifications.

3.1 SVM Parameters

Features are calculated using GLCM before being normalized. GLCM features are computed at eight levels, three distances of one, three and five with four directions of [0 1], [-1 1], [-1 0], [-1 -1] since the measure of co-occurrence obtained at angle 0 would be equal to 180, 45 equal to 225, etc. The distances were chosen based on empirical studies that had shown that values between 1 and 10 give the best accuracy. The values around 1 and 2 had shown highest accuracy. We had chosen 1, 3 and 5 to capture patterns within large distance, yet not too large so that the GLCM wouldn't capture detailed texture information. Thirteen different features for each distance and direction is calculated in addition to the mean of the four directions at each distance. SVM is then trained with C of 2048 and gamma of 0.0313.

3.2 CNN architecture for texture classification

Network architecture design was achieved after considering several architectural designs including AlexNET. The final design was reached after a series of trial and error experiments. The final architecture constitutes the common double Conv-ReLU-Pool then a series of Conv-ReLU since proceeding with pooling layers would cause too much information loss. Figure 21 shows the design details of the architecture.

The structure is designed in a way that each layer has a specific deliverable. For instance, the first layer learns basic patterns of input images. The filter size of the first layer plays a significant role in the upcoming layers up in the hierarchy. In which the first layer captures the repeatable patterns to be further refined up in the hierarchy, so the size of the filter should be able to grasp these patterns. The choice of filter size is made with regard to input image size and expected average size of any pattern. The filter size of later layers is also updated to respect the spatial input size. If the filter size is too small, it might fail to capture a full, complete pattern, and if it is too large, it might capture much noise. Each pooling layer contributes to the network's ability to deal with invariances as they go up in the hierarchy. Because of pooling, the neurons receptive fields become bigger. This lead to better ability to integrate the contextual information over the large spatial area while the fully connected layer could detect the pattern after translation.

The output fully connected layer role is to classify the input sample to one of the possible choices instructed by the dataset configuration.

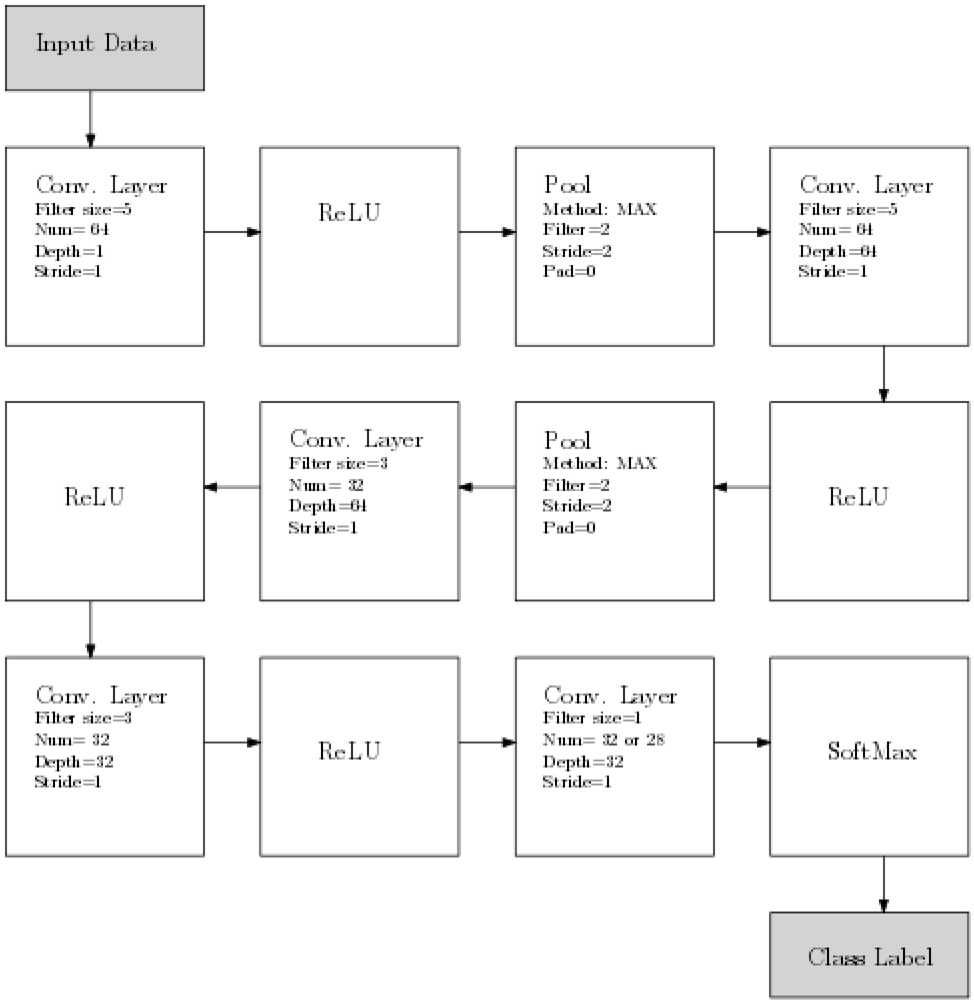


Figure 21 - CNN architecture for texture classification

Initially, the first layer employs 64 filters with the depth of one and filters size 5x5, which is used to grasp the patterns on a very high level. For the CNN to handle variations, the network should capture complete patterns. A moderately sized filter size, receptive field corresponding to input spatial size would allow the network to obtain whole patterns. Choosing the number of filters must be done with caution since a large number of filters may improve accuracy due to the stacking of filters to form receptive fields, but that will also increase the time and computational complexity while few of them usually leads to a lack of generalization. Convolutional layers have fixed stride to prevent mistakes in padding calculation, which is employed to preserve the output filter volume spatially. The filter depth should be proportional to a number of input filters from the previous layer for

1-to-1 multiplication while filters number indicates how many filters with the aforementioned filter depth is used. The filter number is a hyperparameter and not enforced, unlike the depth. The pooling layer is used to reduce the computational parameters as the depth increase and provide non-linear in learning.

At the preprocessing phase, mean subtraction is applied to the training data and the means is saved to be used in testing and validation phases. Scaling initial weights of filters by 0.01 is applied to the randomly distributed weights to make sure the weights are not symmetric. Thus, there is no need for bias to be initialized by non-zero number because bias is used to ensure weights have an asymmetrical shape for the gradient to work and ReLU shall be activated initially. The ReLU activation at initial learning stages isn't likely to introduce marginal performance increase.

To determine the appropriate weights and filter sizes used at convolutional layers and downsampling rates; several CNN architectures were tested. Parallel processing was utilized based on Graphics Processing Unit (GPU) to speed up the testing process.

3.3 Fusion approach

3.3.1 Experts fusion

CNN is known for transformational invariance while SVM is known for high separation accuracy. Fusing both classifiers into one can deliver better results compared to either of them could obtain independently. Fusion could be done in parallel or sequentially. The sequential fusion is done by stacking SVM over CNN by removing Softmax layer and replace it with SVM. The parallel method lets each of the classifiers work independently and adds a fusion layer that acts as on/off switch for each classifier based on classifier expected performance with a particular problem. The method consists of two phases

1. Run series of experiments on trained classifiers to detect average detection accuracy for every single class.
2. Train the fusion layer to assign to each classifier the classes that were reported to have the highest score in accuracy testing.

The process flow is shown in Figure 22. We start by splitting the dataset into four portions. Each part contains a randomly equal number of samples belonging to the same class, so no class has a number of samples that exceeds another in any of the four portions.

The Training set is used for training both classifiers CNN and SVM while the validation set is utilized only by CNN to check for an appropriate number of epochs before stopping the training and prohibit overfitting.

The test set is used to monitor the performance of each classifier for a particular class and trains the fusion layer that assigns the appropriate classifier for given class. The assignment is done by averaging the statistics over a number of iterations defined empirically. Binary mapping is utilized to actualize the classifier assignment.

$$\chi_i = \begin{cases} 1, & A_i^{CNN} >= A_i^{SVM} \\ 0, & otherwise \end{cases}$$

(Eq. 9)

.

The prediction is made such that $\chi_i = 1$ indicates CNN superiority, and appropriateness for usage with the input test sample, otherwise SVM shall be used. Thus, CNN is used when SVM performance is lower than CNN and vice versa. At the last stage, the untouched testing set is used for final assessment of the performance of the fused classifiers. Occasionally for some testing samples, the results of both classifiers votes are within the area of specialization of the classifiers, so confusion matrix tools are used to determine the accuracy of each vote. An equation is used to measure confidence depending on the history of misclassifications for the classes chosen by each classifier. For each test sample I. F(i) = MisclassificationRateCNN(i) - AccuracyRateCNN(i) - MisclassificationRateSVM(i) >= MisclassificationRateSVM(i) - AccuracyRateSVM(i) - MisclassificationRateCNN(i) to determine vote confidence of particular class based on the error rate.

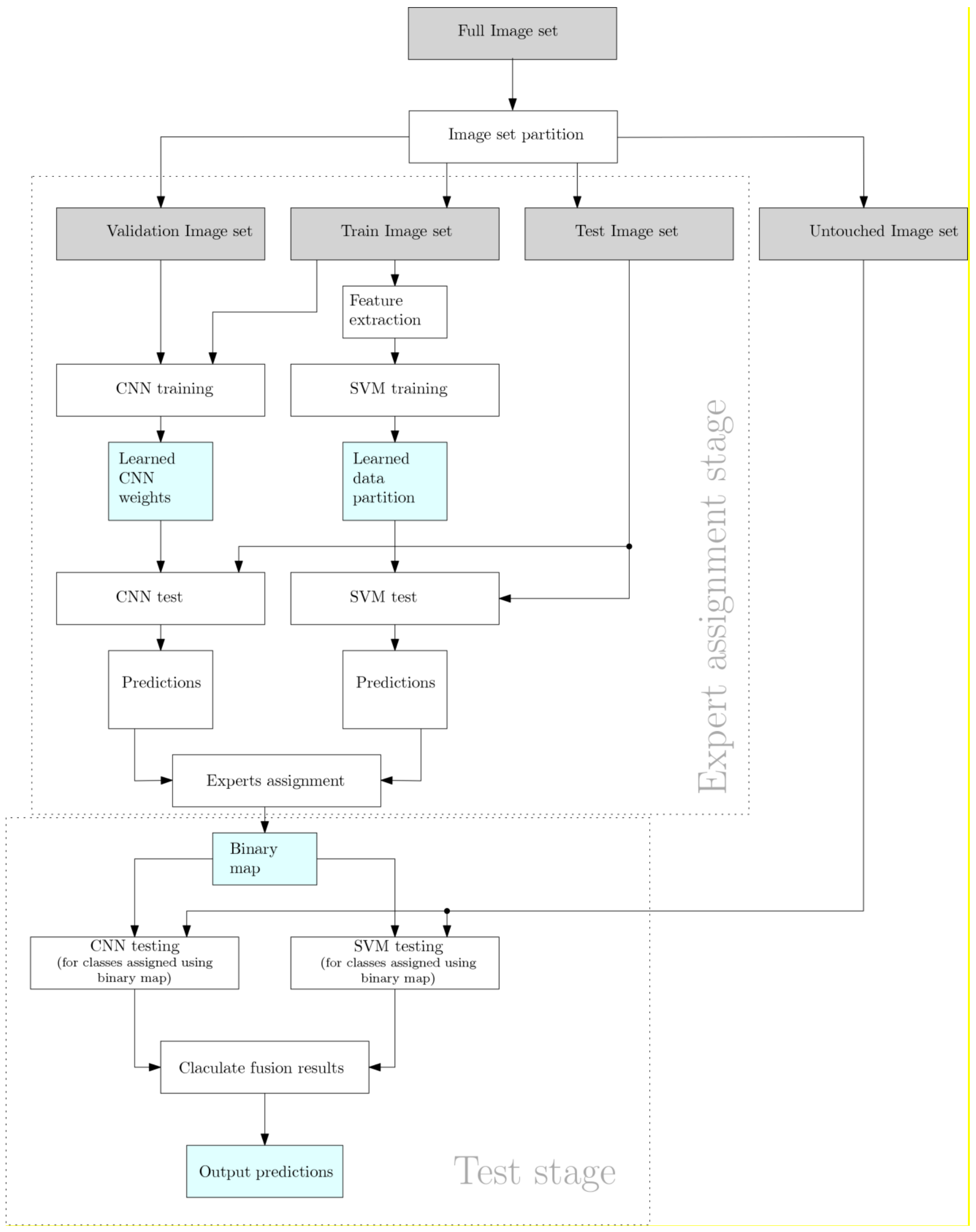


Figure 22 - Basic fusion chart

CHAPTER 4. EXPERIMENTAL RESULTS AND ANALYSIS

4.1 Datasets

4.1.1 Brodatz 32

To analyze the real data, we have chosen Brodatz and Kylberg texture datasets. The Brodatz textures are de-facto standard and widely used as a benchmark dataset in texture segmentation and classification. It consists of 112 textures that were abstracted from the Brodatz texture album [21]. Each of these textures is produced from a single image scanned from the texture album. Figure 23 illustrates the example of several texture samples from this album.

Brodatz32 contains 32 texture classes where each sample is 64×64. Each one of the 32 classes has 64 samples, 16 of them are unique while others are variations of the unique 16. The variations are transformations such are rotation, scaling, or both.

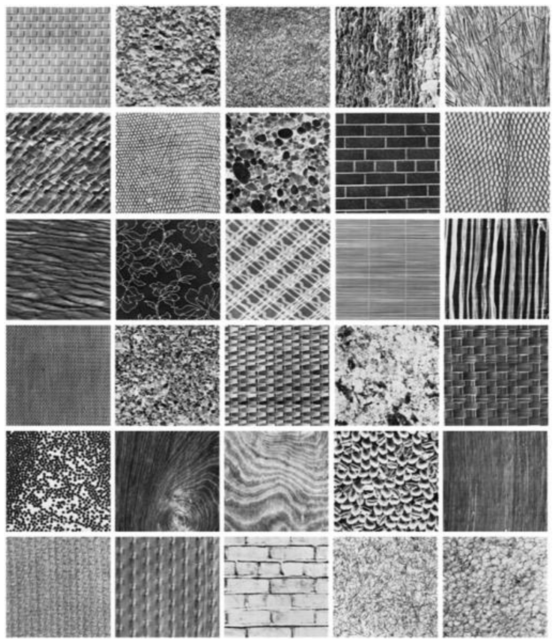


Figure 23 - Samples from Brodatz texture database.

4.1.2 Kylberg

Another great textures dataset is Kylberg [22]. What makes Kylberg of interest is that the imaging conditions are not ideal, and artificial correction using photo editing tools were introduced. Thus, that presents another challenge for classifiers. Figure 24 illustrates several examples from Kylberg texture database.

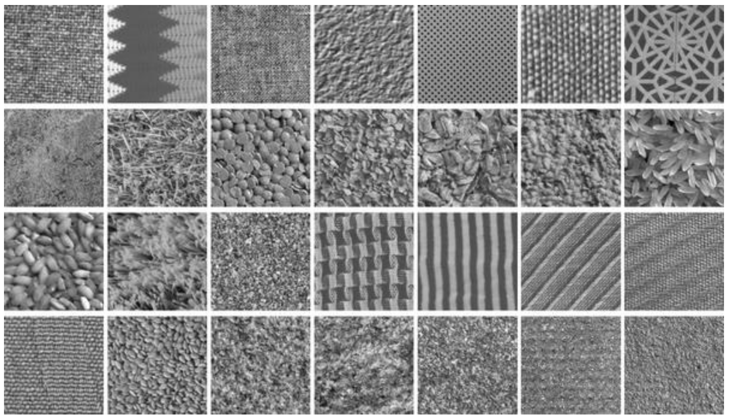


Figure 24 - Texture samples from each class of Kylberg

Kylberg [50] has 28 classes where each of them has 160 samples of size 567x567. Kylberg images were resized to 64 x 64 to make it compatible with the CNN architecture designed to work with moderately sized images in dimension.

4.2 Fusion Parameters Tuning

The fusion binary mapping layer has one primary parameter that requires training. Thus, the fusion is affected by data distribution among training, and testing. Several experiments had been conducted to estimate the best allocation of data for each dataset. For both datasets, experimentation for the singled out parameter was done with a constant ratio of distribution to rest of the data among other parameters.

The training data had been iteratively increased for both datasets to test the effect of training increase on the fused classifier performance. The rest of the parameters

(Validation, Mapping, and test) got data in the ratio of 1-1-2 that is to give testing the highest priority. The constant rate of distribution attempt to freeze other parameters and focus on the training data percentage increase. For Brodatz dataset, the training parameter was tested and returned results shown in Table 1.

Training Percentage	Classification Accuracy
10%	74.66%
20%	79.79%
30%	90.62%
40%	92.70%
50%	85.74%
60%	93.35%
70%	93.75%
80%	91.40%

Table 1 – Training data percentage accuracy for Brodatz dataset

It was concluded from the first experiment that around 60% of the data for training are sufficient for reasonable performance. Similar testing was conducted for Kylberg dataset and returned results shown in Table 2.

Training Percentage	Classification Accuracy
10%	95.98%
20%	98.71%
30%	98.80%
40%	98.54%
50%	99.25%
60%	99.55%
70%	98.88%

Table 2 - Training data percentage accuracy for Kylberg dataset

It was concluded that around 50 percent to 60 percent of the data for training should be sufficient. The marginally higher results for Kylberg dataset than for Brodatz at low training data is attributed to a low similarity between different classes. Thus, the easier separation between its texture, unlike Brodatz that has texture types that look very similar.

The second parameter tested is mapping or fusion training parameter. The testing involved the same methodology used for classifier training test. The training data increased iteratively by 10 percent per iteration. The remaining data was distributed in the ratio of 2-

1-1 on training, validation, and testing giving emphasis on classifier training to fusion training relation. The distribution was done under the assumption that poor classifier training will result in a bad fusion regardless of how proficient the fusion algorithm might appear to be.

The mapping parameter was tested for both classifiers as well. The results for Brodatz are as shown in Table 3.

Training Percentage	Classification Accuracy
10%	90.62%
20%	90.36%
30%	87.23%
40%	89.45%
50%	86.71%
60%	90.62%
70%	89.84%
80%	57.03%

Table 3 – BinaryMap accuracy per percentage of training data for Brodatz dataset

The results show that the increase of training data would have a positive effect on fusion but taking into consideration that dedicating much of the data to fusion layer training would prohibit optimal training of CNN and SVM. Thus, an inversely proportional relation between the classifier training to fusion training. It is noticed that an abrupt increase in fusion training data portion, leads to a big decrease in classifier training performance, causing a high instability in training. Moreover, the retraining of classifiers and number of iterations given the amount of data, which was constant in this case cause random increases, or decreases of classification results. The lower the data and a constant high number of epochs may lead to overfitting. The results of mapping parameter for Kylberg dataset are shown in Table 4.

Training Percentage	Classification Accuracy
10%	99.44%
20%	99.44%
30%	98.77%
40%	98.88%
50%	98.88%
60%	98.88%

70%	98.87%

Table 4 - BinaryMap accuracy per percentage of training data for Kylberg dataset

The mapping parameter wasn't profoundly affected by small training data due to the SVM efficiency of separation under low training data. In most of Kylberg dataset results, the SVM results were more stable and accurate than CNN. Thus, for the Kylberg dataset in specific, the fusion parameter amount of training data wasn't of a concern compared with training data.

The adopted percentage of data distribution for Kylberg given the results obtained above was 60% for training data, 10% for validation, 20% for fusion, and 10% for testing. The distribution for Brodatz was 60% for training, 10% for validation, 20% for fusion, and 10% for testing, which suggests that a generalized amount of training to test data percentages had been reached.

4.3 Brodatz Results Analysis

Throughout the experiments performed, the percentage of training set was the primary factor of classifier performance. At a particular threshold, additional training data don't show significant major improvement in testing results. An initial experiment is made using Brodatz dataset to compare the performance of both classifiers with same training data with different percentages. The testing sample remained constant at 25% of the total amount of data. The result is shown in Figure 25. The decline in CNN performance over time could be contributed to need for more training epochs due to training data increase while the number of training epochs remained 100.

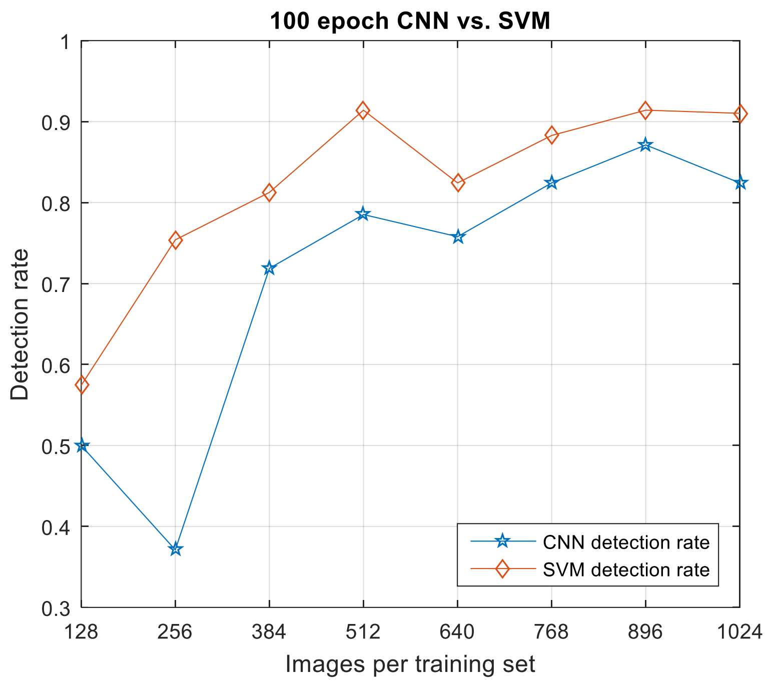


Figure 25 – CNN to SVM accuracy comparison

4.3.1 SVM classification performance using different feature types

Features extractors of texture data, such as LBP and GLCM are frequently used with texture problems. They were compared to other methods, and their results were comparably better regarding classification results and running time [8]. Initial experiments were made to test the performance of both. The SVM's hyper-parameters were set to default configuration defined by LIBSVM during initial testing. The results in Table 5 were expected due to the normalization phase of the training and testing data that were proven to improve GLCM results [53]. GLCM also was shown to be superior to other methods in texture analysis [9].

Dataset	Feature Extractor	Initial Success Rate
Kylberg	LBP	56 %
Kylberg	GLCM	76 %

Kylberg	GLCM + BLP	73 %
Brodatz	LBP	3 %
Brodatz	GLCM	96 %
Brodatz	GLCM + BLP	22 %

Table 5 - Initial feature extractor experimentation

Using GLCM after tuning the Cost and Gamma of SVM showed considerable improvement as shown in Table 6.

No.	1	2	3	Average
Accuracy	98.43%	92.96%	94.53 %	95.30 %

Table 6 - SVM Brodatz training results and average

4.3.2 Analysis of CNN classification accuracy

Preparation for training and testing is carried out as follows for Brodatz dataset

- Training image set: 60 %;
- Validation image set: 10 %;
- Fusion & Testing image set: 30%.

The training set is shared between SVM and CNN while CNN only utilizes the validation. The test set is composed of two sections. Initial testing, which determines the fitness of each classifier and contributes to fusion training. The final testing is fusion testing. Partition is done so that each of the partitions has an equal number of samples belonging to the same class.

No.	1	2	3	Average
Accuracy	98.43%	97.65 %	91.40 %	95.82 %

Table 7 - CNN Brodatz training results and average

4.3.3 Results of fusion scheme

The fusion uses a binary map trained on obtained data from each classifier after training and testing independently. The classifiers are tested with the same independent testing set to determine the fitness of each classifier on the same input sample. That facilitates measuring the statistical gain on each classifier performance using the random class representatives. Figure 26 shows Misclassification rates over ten iterations.

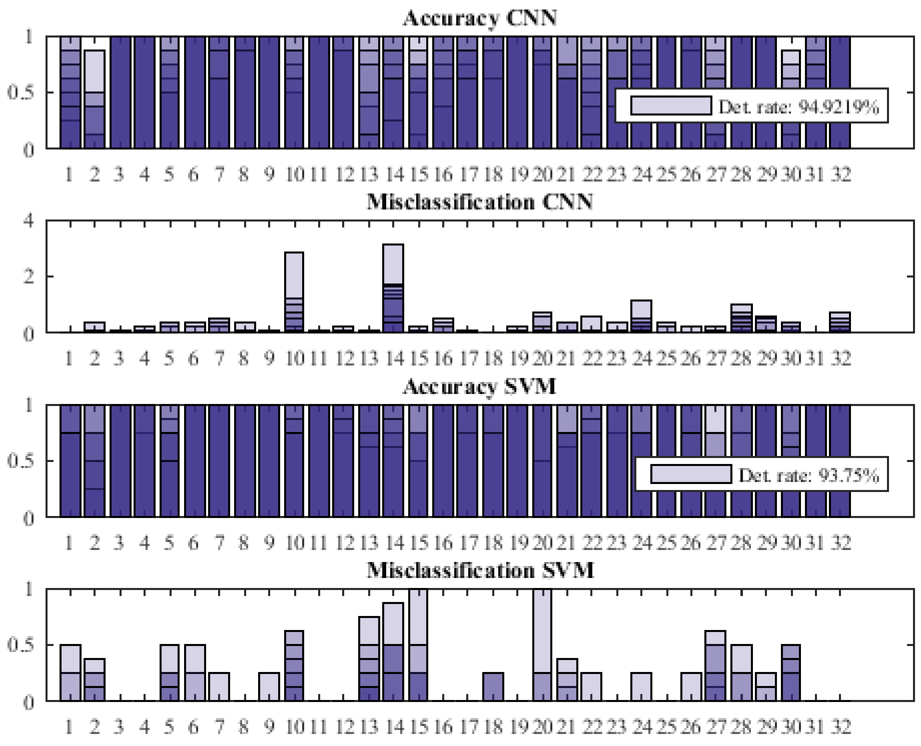


Figure 26 - Experts assignment stage per class accuracies

Figure 27 shows performance recorded for a full test sample. The first two diagrams show the prediction performance on the test samples for each classifier independently, and thirds show the result after using the proposed fusion method. The coloring is used to distinguish instances classified using SVM or CNN. The Red indicates classification using CNN while another color refers to SVM.

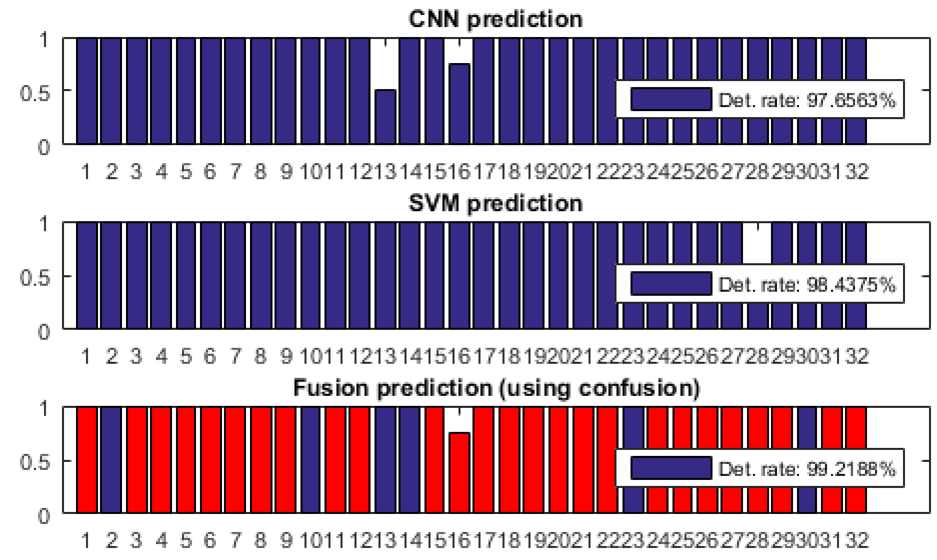


Figure 27 - Bare CNN, SVM, and fusion results

No.	1	2	3	Average
Accuracy	98.43%	99.21 %	96.87 %	98.17 %

Table 8 – Brodatz Fusion experimental results

4.3.4 Results Comparison

Brodatz is a heavily used dataset in scientific research. Datasets like Kylberg had seen researchers reaching 100% classification accuracy. In our study, several experiments were conducted to determine the average, worst, best possible performance that could be obtained from the fusion. It was found that the fusion algorithm reached 99.21%, which wasn't seen at any research before. Although on the other hand, the random initial weights, play a significant factor in achieving optimal results. Even with same training set, the results on CNN may be entirely different from previous training attempts. On the other hand, SVM is relatively stable due to different mechanism employed for training and obtaining margin. It's worth noting that choosing such setup allowed for fault tolerance to take place when CNN training failed to recognize the considerable amount of testing set samples, SVM was able to take over and obtain acceptable fusion results. The CNN achieved a testing accuracy of 91.40% on the third experiment, which shows the effect of training on different samples of data and initial weights.

Algorithm	Accuracy
Gabor Filters (KNN) [44]	95.8%
Multiple texture descriptors (Decision Trees) [45]	96.5%
Dominant LBP (KNN) [44]	98.30%
Robust LBP (KNN) [44]	98.90%
SVM (RZM) [43]	**80%**
1NN (GP-criptor) [47]	**96.3%**
CNN [2]	91.27%
Proposed Method (CNN + SVM)	**98.17 % (+- 1.3)**

Table 9 - Results comparison to other methods for Brodatz dataset classification

4.4 Kylberg Results Analysis

The same approach was used on Kylberg dataset discussed above. The data set is well suited for CNN training using the same proposed network architecture.

4.4.1 CNN & SVM Performance

The same CNN architecture was used successfully with Kylberg due to similar image size after resize; the same filter sizes were able to perform same operations on Kylberg training samples, and relatively similar results to CNN.

No.	1	2	3	Average
Accuracy	97.32%	95.53%	95.31%	96.05%

Table 10 - CNN Kylberg training results and average

SVM was able to obtain superior results to CNN since Kylberg doesn't' introduce transformations, unlike Brodatz. Thus, it shall be easy to extract features, but classification is the key task. CNN may have failed to achieve similar results due to poor classifier compared to efficient marginal separation employed by SVM.

No.	1	2	3	Average
Accuracy	100%	99.33%	99.55%	99.62%

Table 11 - SVM Kylberg training results and average

4.4.2 Fusion Performance

It's surprising that CNN wasn't able to provide any support or enhancement to final fusion results given that nearly optimal accuracy for both classifiers. SVM showed superiority in all class classification.

No.	1	2	3	Average
Accuracy	100%	99.33%	99.55%	99.62%

Table 12 – Kylberg Fusion experimental results

4.4.3 Results Comparison

The results obtained were very competitive, with 0.38% misclassifications from the optimal. It's worth noting that in our initial testing of SVM using GLCM feature extractor, we were able to obtain 70% classification accuracy. After input resizes and optimization, the result was 99.62%, which might explain the low classification accuracy obtained by another researcher on using SVM along with GLCM.

Algorithm	Accuracy
KNN (nLBPd) [48]	99.64%
1NN (GP-Criptor) [47]	**89.5%.**
SVM (RZM) [43]	**99%**
NNge (LBP) [47]	88.26%
SVM (GLCM) [47]	79.99 %
1-NN [49]	99.7 %
CNN (AlexNet) [46]	99.4 %
Proposed Method (CNN + SVM)	**99.62 % (+- 0.29)**

Table 13 - Results comparison to other methods for Kylberg dataset classification

4.5 Discussion

Zernike moments is a commonly used method in pattern recognition but rarely seen in texture analysis. Ida-Maria et al. [43] proposed adapting the method to be used with textures. They introduced regional Zernike moment (RZM) that provides statistical patterns similar to pattern extractors used in their research GLCM, LBP, etc. The method successfully approached 100% classification rate on Kylberg dataset, but failed to obtain similar results with Brodatz dataset.

Harith Al-Sahaf et al. [47] proposed using Genetic Programming (GP) for feature extraction that works by generating a descriptor from only two training instances per class. They have tested the descriptors with multiple classifiers and scored relatively higher results for Brodatz dataset at 96.3% compared to Kylberg at 89.5%.

The proposed methodology scores exceed both methods delivering results approaching 99% - 100% for both datasets. Although regional Zernike moments scored similar results to the ones obtained by the proposed fused classifier with Kylberg dataset, it failed to obtain similar results with Brodatz dataset. Similarly, 1NN(GP) obtained 96.3% successful classification rate on Brodatz dataset which is relatively close to 98.1% obtained by the proposed method, the 1NN(GP) failed to maintain consistency and stability in the result for the Kylberg dataset obtaining 89.5% while the proposed method obtained 99.6%.

On the downside, we have seen a dramatic increase in the time required due to the training of two classifiers, but the increase in classification rates is the tradeoff to extra time paid during training.

CHAPTER 5. CONCLUSIONS

5.1 Contributions

Both CNN and SVM have different learning approaches. Their fusion would usually lead to improvement in classification rates. The fusion of SVM superior margin separation to CNN deep learning ability to handle transformations could potentially be the solution to problems that is complex in the nature. In some cases, one classifier may be superior to the other, in that case, fusion would still ensure a reasonable final classification result regardless of one of the classifiers lacking in accuracy due to the nature of the problem.

5.2 Future Work

The training time shall be improved in the future by implementing fully parallelized solution instead of the only parallelization of the inner-working of both classifiers. This is possible due to the nature of fusion at very late stage of training at the fusion layer. Data distribution could also be improved by sharing part of the training data of the classifiers to train the fusion layer, although sharing the same data between the fusion layer and classifiers is not likely to produce acceptable results, sharing part of the data may allow the fusion layer to exist without worrying about data share consumption used by the layer.

REFERENCES

1. Y. Bengio, Learning deep architectures for AI, Foundations and Trends in Machine Learning, Vol. 2, No. 1, 2009, pp. 1-127.
2. L. Hafemann, An analysis of deep neural networks for texture classification, a Master thesis, 2014, 89p.
3. T. Xu and I. Gondra, “Texture map: An Effective Representation for Image Segmentation,” Proc. 2009 C3S2E Conf. - C3S2E ’09, p. 197, 2009.
4. A. Song, Texture Classification: A Genetic Programming Approach. Ph. D. Thesis, School of Computer Science and Information Technology, Melbourne, Australia, 2003.
5. J. Heaton, Artificial Intelligence for Humans: Deep Learning and Neural Networks, 2015.
6. Chang, C., Lin, C-J.: LIBSVM: a library for a support vector machines http://www.csie.ntu.edu.tw/~cjlin/libsvm (2001). Accessed 26 August 2015
7. C.-M. Chen ; C.-C. Chen ; C.-C. Chen, A Comparison of Texture Features Based on SVM and SOM, International Conference on Pattern Recognition (ICPR), 2006, pp. 630-633.
8. B. Salem, S. Nasri, Rotation Invariant Texture Classification using Support Vector Machines Int. Conf. on Communications, Computing and Control Applications, 2011, pp. 1-6.
9. S. Suralkar, A. Karode, P. Pawade, Texture image classification using support vector machine, Int. Journal of Comp. Technologies, Vol. 2, No. 1, 2012, pp. 71-75.
10. K.Rajpoot, N. Rajpoot, Wavelets and Support Vector Machines for Texture Classification, Proc. On Int. Multitopic Conference, 2004, pp. 328 – 333.
11. R. M. Haralick, K. Shanmugam, and I. Dinstein. Textural features for image classification. IEEE Trans. on SMC, SMC-3(6):610–621, Nov. 1973.
12. Liao, M. Law, A. Chung, Dominant Local Binary Patterns for Texture Classification, IEEE Trans. On Image Processing, Vol. 18, No. 5, 2009, pp. 1107-1118.

13. T. Olaya, M. Pietikainen, Unsupervised texture segmentation using feature distributions, Pattern Recognition, Vol. 32, 1999, pp. 477-486.

14. . Grigorescu, N. Petkov, P. Kruizinga, Comparison of Texture Features Based on Gabor Filters, IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 11, NO. 10, OCTOBER 2002, pp. 1160-1167.

15. S.-M. Pun, H.-M. Zhu, Textural Image Segmentation Using Discrete Cosine Transform, PROCEEDINGS OF THE 3RD INTERNATIONAL CONFERENCE ON COMMUNICATIONS AND INFORMATION TECHNOLOGY, 2009, pp. 54-58.

16. G. Sorwar, A. Abraham, DCT based texture classification using a soft computing approach. Malaysian Journal of Computer Science, vol. 17 No. 1, pp. 13-23, June 2004.

17. X. Liu, D. Wang, Texture Classification Using Spectral Histograms, IEEE Trans. On Image Processing, Vol.12, No. 6, 2003, pp. 661-670.

18. Jiangye Yuan, DeLiang Wang, and Rongxing Li, Image Segmentation Based on Local Spectral Histograms and Linear Regression. Proceedings of International Joint Conference on Neural Networks, San Jose, California, USA, 2011.

19. X. Liu, D. Wang, A spectral histogram model for texton modeling and texture discrimination, Vision Research, Vol. 42, 2002, pp. 2617–2634.

20. E. Hadjidemetriou , M. D. Grossberg , S. K. Nayar , Multiresolution Histograms and their Use for Texture Classification, International Workshop on Texture Analysis and Synthesis, 2003.

21. P. Brodatz, Textures: A photographic album for artists and designers, Dover, New York, 1996.

22. G. Kylberg, Kylberg texture dataset, Centre of image analysis, Swedish University of Agricultural Sciences, 2011.

23. K.J. Dana, B. Van-Ginneken, S.K. Nayar, J.J. Koenderink, "Reflectance and Texture of Real World Surfaces," ACM Transactions on Graphics (TOG), Vol.18, No.1, pp.1-34, Jan, 1999.

24. F. Liu, W. Picard, Periodicity, directionality and randomness: Wold features for image modeling and retrieval, IEEE Trans. On PAMI, Vol. 18, No. 7, 1996, pp. 722-733.

25. A. Materka, M. Strzelecki, Texture Analysis Methods – A Review, Technical University of Lodz, Institute of Electronics, COST B11 report, Brussels 1998

26. J. Mridula and D. Patra, “Utilization of Grey Level Co-occurrence Matrix and Markov Random Field Model for Segmentation of Colour Textured Images,” Int. Conf. Commun. Comput. Secur., pp. 421–426, 2011.

27. M. Mehri, P. Gomez-Krämer, P. Héroux, A. Boucher, and R. Mullot, “Texture feature evaluation for segmentation of historical document images,” Proc. 2nd Int. Work. Hist. Doc. Imaging Process. - HIP ’13, p. 102, 2013.

28. B. M. Carvalho, T. S. Souza, and E. Garduño, “Texture fuzzy segmentation using adaptive affinity functions,” Proc. 27th Annu. ACM Symp. Appl. Comput. - SAC ’12, p. 51, 2012.

29. A. Dixit and N. P. Hegde, “Image Texture Analysis - Survey,” 2013 Third Int. Conf. Adv. Comput. Commun. Technol., pp. 69–76, 2013.

30. J. Raju and C. A. D. Durai, “A survey on texture classification techniques,” 2013 Int. Conf. Inf. Commun. Embed. Syst. ({ICICES)}, pp. 180–184, 2013.

31. Z. Guo, L. Zhang, and D. Zhang, “A completed modeling of local

binary pattern operator for texture classification,” IEEE Trans. Image Process., vol. 19, no. 6, pp. 1657–1663, Jun. 2010.

32. S. Liao, M. W. K. Law, and A. C. S. Chung, “Dominant local binary patterns for texture classification,” IEEE Trans. Image Process.,vol.18, no. 5, pp. 1107–1118, May 2009.

33. C. S. Chung, Shu Liao and Albert, “Texture Classification by Using Advanced Local Binary Patterns and Spatial Distribution of Dominant Patterns,” PAMI, vol. 8, no. 7, pp. 1221–1224, 2007.

34. Fakhry M. Khellah,"Texture Classification Using Dominant Neighborhood Structure,"IEEE Trans. Pattern analysis, vol. 20, no. 11, pp. 3270–3278, November. 2011.

35. C. Tanchotsrinon, S. Phimoltares, and C. Lursinsap, "An autonomic building detection method based on texture analysis, color segmentation, and neural classification," *2013 5th Int. Conf. Knowl. Smart Technol.*, pp. 162–167, 2013.

36. J. Melendez, D. Puig, and M. Angel Garcia, "On adapting pixel-based classification to unsupervised texture segmentation," Proc. - Int. Conf. Pattern Recognit., pp. 854–857, 2010.

37. S. Rathore, M. A. Iftikhar, M. Hussain, and A. Jalil, "Texture Analysis for Liver Segmentation and Classification: A Survey," 2011 Front. Inf. Technol., pp. 121–126, 2011.

38. a P. Nanthagopal and R. Sukanesh, "Wavelet statistical texture features-based segmentation and classification of brain computed tomography images," {IET} Image Process., vol. 7, no. 1, pp. 25–32, 2013.

39. S. Masoudnia and R. Ebrahimpour, "Mixture of experts: A literature survey," Artif. Intell. Rev., vol. 42, no. 2, pp. 275–293, 2014.

40. A. Rampun, H. Strange, and R. Zwiggelaar, "Texture segmentation using different orientations of GLCM features," Proc. 6th Int. Conf. Comput. Vis. / Comput. Graph. Collab. Tech. Appl. - MIRAGE '13, 2013.

41. Y. Bengio, "Learning deep architectures for AI," Foundations and trends in Machine Learning, vol. 2, no. 1, p. 1127, Jan. 2009.

42. S. Hossain and S. Serikawa, "Texture databases - A comprehensive survey," Pattern Recognit. Lett., vol. 34, no. 15, pp. 2007–2022, 2013.

43. I. Sintorn, "Regional Zernike Moments for Texture Recognition," Int. Conf. Pattern Recognit., no. Icpr, pp. 1635–1638, 2012.

44. J. Chen, V. Kellokumpu, G. Zhao, and M. Pietikinen, "RLBP: robust local binary pattern," in Proceedings of the British Machine Vision Conference. BMVC, 2013.

45. E. Urbach, J. Roerdink, and M.Wilkinson, "Connected shape-size pattern spectra for rotation and scale-invariant classification of gray-scale images," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, no. 2, pp. 272–285, 2007.

46. V. Andrearczyk and P. F. Whelan, "Using Filter Banks in Convolutional Neural Networks for Texture Classification," Pattern Recognit. Lett., p. 12, 2016.

47. H. Al-Sahaf, M. Zhang, M. Johnston, and B. Verma, “Image descriptor: A genetic programming approach to multiclass texture classification,” 2015 IEEE Congr. Evol. Comput. CEC 2015 - Proc., pp. 2460–2467, 2015.

48. Y. Kaya, Ö. Faruk, E. Grul, and R. Tekin, “Two novel local binary pattern descriptors for texture analysis,” Appl. Soft Comput. J., vol. 34, pp. 728–735, 2015.

49. P. S. Hiremath and R. A. Bhusnurmath, “DIFFUSION APPROACH FOR TEXTURE ANALYSIS BASED ON,” vol. IX, no. Vii, pp. 108–121, 2015.

50. G. Kylberg. The Kylberg Texture Dataset v. 1.0, Centre for Image Analysis, Swedish University of Agricultural Sciences and Uppsala University, External report (Blue series) No. 35. Available online at: http://www.cb.uu.se/~gustaf/texture/

51. D. Scherer, A. Mller, and S. Behnke, “Evaluation of pooling operations in convolutional architectures for object recognition,” in Artificial Neural NetworksICANN 2010. Springer, 2010, p. 92101.

52. Dropout: A Simple Way to Prevent Neural Networks from Overfitting Nitish Srivastava nitish@cs.toronto.edu Geoffrey Hinton hinton@cs.toronto.edu, Alex Krizhevsky kriz@cs.toronto.edu Ilya Sutskever ilya@cs.toronto.edu, Ruslan Salakhutdinov

53. Y. Ben Salem and S. Nasri, “Automatic recognition of woven fabrics based on texture and using SVM,” Signal, Image Video Process., vol. 4, no. 4, pp. 429–434, 2009.

54. F. J. Huang and Y. LeCun, “Large-scale learning with SVM and convolutional nets for generic object categorization,” Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., vol. 1, pp. 284–291, 2006.