IDENTIFICATION OF BUTTERFLY SPECIES USING MACHINE LEARNING AND IMAGE PROCESSING TECHNIQUES

2020
DOCTOR OF PHILOSOPHY
COMPUTER ENGINEERING

Ayad Saad ALMRYAD

Assist. Prof. Dr. Hakan KUTUCU

IDENTIFICATION OF BUTTERFLY SPECIES USING MACHINE LEARNING AND IMAGE PROCESSING TECHNIQUES

Ayad Saad ALMRYAD

T.C.
Karabuk University
Institute of Graduate Programs
Department of Computer Engineering
Doctor of Philosophy

Assist. Prof. Dr. Hakan KUTUCU

KARABUK

February 2020

I certify that in my opinion, the thesis submitted by Ayad Saad ALMRYAD titled "IDENTIFICATION OF BUTTERFLY SPECIES USING MACHINE LEARNING AND IMAGE PROCESSING TECHNIQUES" is fully adequate in scope and in quality as a thesis for the degree of Doctor of Philosophy.

Assist. Prof. Dr. Hakan KUTUCU .... [illegible]
Thesis Advisor, Department of Computer Engineering

This thesis is accepted by the examining committee with a unanimous vote in the Department of Computer Engineering as a Doctor of Philosophy thesis. 14/02/2020

Examining Committee Members (Institutions)	Signature
Chairman: Assoc. Prof. Dr. Sedat AKLEYLEK (OMÜ)	[illegible]
Member : Assoc. Prof. Dr. Oğuz FINDIK (KBÜ)	[illegible]
Member : Assist. Prof. Dr. Salih GÖRGÜNOĞLU (KÜ)	[illegible]
Member : Assist. Prof. Dr. Muhammed Kamil TURAN (KBÜ)	[illegible]
Member : Assist. Prof. Dr. Hakan KUTUCU (KBÜ)	[illegible]

28/02/2020

The degree of Doctor of Philosophy by the thesis submitted is approved by the Administrative Board of the Institute of Graduate Programs, Karabük University.

Prof. Dr. Hasan SOLMAZ .... [illegible]
Director of the Institute of Graduate Programs

“I declare that all the information within this thesis has been gathered and presented in accordance with academic regulations and ethical principles and I have according to the requirements of these regulations and principles cited all those which do not originate in this work as well.”

Ayad Saad ALMRYAD

ABSTRACT

Ph. D. Thesis

IDENTIFICATION OF BUTTERFLY SPECIES USING MACHINE LEARNING AND IMAGE PROCESSING TECHNIQUES

Ayad Saad ALMRYAD

Karabük University
Institute of Graduate Programs
Department of Computer Engineering

Thesis Advisor:
Assist. Prof. Dr. Hakan KUTUCU
February 2020, 70 pages

In today's competitive conditions, producing fast, inexpensive and reliable solutions are objectives for engineers. Development of artificial intelligence and the introduction of this technology to almost all areas have created a need to minimize the human factor by using artificial intelligence in the field of image processing, as well as to make a profit in terms of time and labor. In this thesis, we propose an automated butterfly species identification model using artificial neural and deep neural networks.

The study in the thesis consists of two stages. In the first stage, we studied on lab-based butterfly images taken on under a fixed protocol. The species of butterflies in these images are identified by expert entomologists. We used a total of 140 images for lab-based butterfly images of 10 species. After applying some preprocess to the images such as histogram equalization and background removing, we extracted several features from the butterfly images. Finally, we used an artificial neural network in MATLAB version R2014b using the Neural Network Toolbox for butterfly identification. The ANN model achieved an accuracy of 98%.

In the second stage of the thesis, we studied on field-based butterfly images. We collected 44659 images of 104 different butterfly species taken with different positions of butterflies, the shooting angle, butterfly distance, occlusion, and background complexity in the field in Turkey. Since many species have a few image samples we constructed a field-based dataset of 17769 butterflies with 10 species. Convolutional Neural Networks (CNNs) implemented by Python were used for the identification of butterfly species. Comparison and evaluation of the experimental results obtained using three different network structures are conducted. Experimental results on 10 common butterfly species showed that our method successfully identified various butterfly species.

Key Words : Artificial intelligence, deep learning, computer vision, butterfly species recognition, feature extraction, feature selection.

Science Code : 92431

ÖZET

Yüksek Lisans Tezi

MAKİNE ÖĞRENMESİ VE GÖRÜNTÜ İŞLEME TEKNİKLERİ KULLANILARAK KELEBEKLERİN TANIMLANMASI

Ayad Saad ALMRYAD

Karabük Üniversitesi
Lisansüstü Eğitim Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Tez Danışmanı:
Dr. Öğr. Üyesi Hakan KUTUCU
Şubat 2020, 70 sayfa

Günümüzün rekabetçi koşullarında, hızlı, ucuz ve güvenilir çözümler üretmek mühendisler için bir hedeftir. Yapay zekanın geliştirilmesi ve bu teknolojinin neredeyse tüm alanlara tanıtılması, görüntü işleme alanında yapay zekayı kullanarak insan faktörünü en aza indirmenin yanı sıra zaman ve emek açısından kar elde etme ihtiyacını yarattı. Bu tezde, yapay sinir ve derin sinir ağlarını kullanarak otomatik bir kelebek türü tanımlama modeli öneriyoruz.

Tezdeki çalışma iki aşamadan oluşmaktadır. İlk aşamada, sabit bir protokol altında alınan laboratuvar tabanlı kelebek görüntüleri üzerinde çalıştık. Bu görüntülerdeki kelebek türleri uzman entomologlar tarafından tanımlanmıştır. 10 türün laboratuvar tabanlı kelebek görüntüleri için toplam 140 görüntü kullandık. Histogram eşitleme ve arka plan kaldırma gibi görüntülere bazı ön işlemler uyguladıktan sonra, kelebek görüntülerden çeşitli özellikler çıkardık. Son olarak, kelebek tanımlama için Neural

Network paketi kullanılarak MATLAB R2014b versiyonunda yapay sinir ağı kullandık. YSA modeli % 98'lik bir doğruluk elde etmiştir.

Tezin ikinci aşamasında alan bazlı (doğadan) kelebek görüntüleri üzerinde çalıştık. Türkiye'de doğada çekilmiş farklı pozisyon, çekim açısı, kelebek mesafesi ve arka plan karmaşıklığı ile alınan 104 farklı kelebek türünden 44659 görüntü topladık. Birçok türün birkaç görüntü örneği olduğundan, 10 tür içeren 17769 kelebeğin alan tabanlı bir veri kümesini oluşturduk. Kelebek türlerinin tanımlanmasında Python tarafından uygulanan Evrişimli Sinir Ağları (CNN) kullanılmıştır. Üç farklı ağ yapısı kullanılarak elde edilen deneysel sonuçların karşılaştırılması ve değerlendirilmesi yapılmıştır. 10 yaygın kelebek türü üzerinde yapılan deneysel sonuçlar, yöntemimizin çeşitli kelebek türlerini başarıyla tanımladığını göstermiştir.

Anahtar Kelimeler : Yapay zeka, derin öğrenme, bilgisayarla görme, kelebek türlerinin tanınması, özellik çıkarma, özellik seçimi.

Bilim Kodu : 92431

ACKNOWLEDGMENT

First of all, I would like to express my gratitude to Allah, who has given me these days, always protects me, gives me strength and makes me successful.

I would like to thank Dr. Hakan KUTUCU whom I have benefited from his extensive knowledge and experience in the planning, research, execution, and formation of this thesis. In addition, I would like to thank all of my teachers and colleagues who have always supported me with their help, knowledge, and experience during my undergraduate and graduate education.

Finally, I thank my dear family for their patience, self-sacrifice, and support, and for all physical and spiritual assistance with my whole heart.

CONTENTS

Page

Page

LIST OF FIGURES

Page

Page

LIST OF TABLES

SYMBOLS AND ABBREVIATIONS INDEX

SYMBOLS

gx & gy	: Spatial Gradient
P	: Speckle noise distribution image
I	: the input image
N	: the uniform noise image
T	: the threshold value
R	: the circle of radius
$F1$	: the average of texture features
$F2$	: the deviation of texture features
$F3$	: the energy of texture features
$F4$	: the entropy of texture features
$F5$	: the correlation of texture features
$M \times N$	: the matrix of size
K	: the maximum value for the ULBP matrix
M:	: the mean
σ	: the standard deviation.
p	: the padding size
s	: the stride number
FC	: fully-connected layers
IL	: input layer
CL	: convolution layer
PL	: pooling layer
OL	: output layer

ABBREVIATIONS

RESNET50	:	Deep Residual Learning for Image Recognition
VGG-16 – 19	:	Visual Geometry Group - 16
CNN	:	Convolutional Neural Networks
RGB	:	Red-Green-Blu
HSV	:	Hue-Saturation-Value
ULBP	:	Uniform Local Binary Patterns
LBP	:	Local Binary Patterns
CSLBP	:	Center Symmetric Local Binary Patterns
SIFT	:	Scale-Invariant Feature Transform
ANN	:	Artificial Neural Network
MSE	:	Mean Squared Error
ICA	:	Linear Discriminant Analysis
PCA	:	Principal Component Analysis
SVM	:	Support vector Machines
ED	:	Edge Detection
GM	:	Graph Modeling
JPEG	:	Joint Photographic Experts Group
TIFF	:	Tag Image File Format

PART 1

INTRODUCTION

Butterflies are insects in the order Lepidoptera. The number of butterfly species in the world varies from 15000 to 21000 according to a recent estimate in [1]. Due to various species, high similarity and the characteristics of the distinction of butterflies are not evident, the identification and classification of butterflies have the problems of low accuracy and slow recognition. Furthermore, the number of taxonomists and trained technicians have decreased dramatically.

The discrimination between butterfly species requires expertise and time, which is not always available, but after the development of software that identifies butterfly species by extracting features from images, the need for experts will reduce. There are two main problems in the existing butterfly species identification research based on computer vision techniques. First, collecting the butterfly dataset is difficult, identifying is time-consuming work for entomologists, and the number of butterflies included in the butterfly dataset is not comprehensive. Second, the butterfly pictures used for training are all pattern pictures with obvious morphological features, lacking the ecological pictures of butterflies in nature. Furthermore, the differences between the two pictures are obvious that makes the combination of research and production difficult and the recognition accuracy is low [2]. Therefore, it is of great importance to make research on the automatic identification of butterflies and improve its accuracy and efficiency. The development of tools for automating the identification of butterfly species has an important contribution to the literature.

This thesis is organized as follows: In Part 2, we discuss the theoretical background of insect classification and identification. We introduce image processing techniques related to butterfly segmentation in Part 3. We give a short review of artificial neural network (ANN) and the ANN architecture to identify butterfly species in Part 4. We give an extensive literature review of insect classification researches in Part 5. The field butterfly image dataset and the methodology are discussed in Part 6. We introduce local binary patterns which will be used to extract some features from the butterflies in Part 7. In Part 8, we give the features which are extracted and used as input for the artificial neural network. A technical approach based on the different convolutional neural network architectures is introduced in Part 9. Finally, the conclusion section concludes the thesis.

PART 2

THEORETICAL BACKGROUND

Due to various species, high similitude and the attribute of the divergence is not evident, the identification and classification of butterflies have the complications of low precision and steady identification. Therefore, it is very significant to do research on the automatic recognition of butterflies and better its accuracy and orderliness. The development of tools for automating the identification of butterfly species has an important contribution to the literature. The discrimination between butterfly species requires expertise and time, which is not always available, but after the development of software that identifies some butterfly species by extracting features from butterflies.

There exist two main problems in the be living butterfly species recognition research. Firstly, collecting the butterfly set of information is hard, identifying is time-consuming work for entomologists, and various butterflies comprised in the butterfly set of information is not thorough. Secondly, the pictures of the butterflies geared towards training are all pattern pictures having obvious morphological features, missing the ecological pictures of butterflies in the natural ecological environment. Furthermore, the unlikenesses between two pictures are apparent making the merge of production and experimentation difficult and the identification correctness is low.

Kaya et al. [3] proposed two novel local binary patterns (LPB) descriptors for detecting special textures in images. One is established on the correlation between the sequential neighbors of a center pixel with a defined distance and the other is established on determining the neighbors in the same orientation through the central pixel parameter. They tested their descriptors to identify butterfly species on lab-based images of 140 butterflies collected in Van city of Turkey. The highest accuracy they obtained for classifying butterflies by the artificial neural network is

95.71%. Kaya and Kaycı [4] developed a model for butterfly identification depending on color and texture characteristics by the artificial neural network using the gray level co-occurrence matrix with different angles and distances. The correctness of their method reached 92.85%. Wen and Guyer [5] developed a model which combines local and global features for insect classification via five classifiers that are normal densities based linear classifier (NDLC), the minimum least-square linear classifier (MLSLC), nearest mean classifier (NMC), K-nearest neighbor classifier (KNNC) and decision tree (DT). Their experimental results tested on images obtained from actual field trapping for training yielded the classification rate of 86.6%.

Xie et al. [6] proposed a learning model for the classification of insect images by the use of advanced multiple-task sparse representation and multiple-kernel learning techniques. They tested the proposed model on 24 common pest species of field crops and compared it with some recent methods.

Feng et al. [7] improved an insect species identification and retrieval system established on wing characteristics of moth images. Their retrieval system is based on CBIR architecture not being about having a final answer but just issuing a match-list to the user.

Yao et al. [8] designed a model to automate rice pest identification using 156 features of pests. The authors tested the proposed model on a few species of Lepidoptera rice pests with middle size using the support vector machine classifier.

Another study for pest recognition was done by Faithpraise et al. using the k-means clustering algorithm and correspondence filters [9]. Leow et al. [10] developed an automated identification system of copepod specimens by extracting morphological features and using ANN. They used seven copepod features of 240 sample images and estimated an overall accuracy of 93.13%.

Zhu and Zhang [11] proposed a technique to classify lepidopteran insect images by integrated region matching and dual-tree complex wavelet transform. They tested the method on a database including 100 lepidopteran insects of 18 families and estimated the identification correctness of 84.47%.

Mayo and Watson [12] showed how effective data mining techniques could be for the recognition of species. They used WEKA which is a data mining tool with different classifiers such as Naïve Bayes, instance-based learning, random forests, decision trees, and support vector machines. WEKA was able to achieve an accuracy of 85% using support vector machines to classify live moths by species.

Another work for identification of live moths was conducted by Watson et al [13] who proposed an automated identification tool named DAISY. Silva et al. [14] aimed to investigate the best combination of feature selection techniques and classifiers to identify honeybee subspecies. They found the best pair as the combination of Naïve Bayes classifier and the Correlation-Based feature selector in their experimental results among seven combinations of feature selectors and classifiers.

Wang et al. [15] designed an identification system of insect images at the order level. They extracted seven features from insect images. However, the authors manually removed some attached to insects such as pins. Their method has been tested on 225 specimen images from nine orders and sub-orders using an artificial neural network with an accuracy of 93%.

The discrimination between butterfly species requires expertise and time, which is not always available, but after the development of software that identifies some butterfly species by extracting features from butterflies' wings, the species identification became faster and efficient.

The development of the system to identify insect species needs great efforts. Machine learning techniques like Deep Learning, Convolutional Neural Networks (CNNs), principal component analysis (PCA), linear discriminant analysis (LDA), artificial neural networks (ANNs), support vector machines (SVMs) can be used to create automated identification system.

Pattern recognition and machine learning are jointly studied two branches of artificial intelligence. Pattern recognition use experience gained from improvements in machine learning and image processing techniques. Many studies on pattern recognition include image processing and machine learning. In these systems, image processing techniques are used to identifying the objects in input images and then machine learning is used to learn the system for the change in pattern. With the advent of new methods and techniques on related fields, pattern recognition can be used more and more computer-aided diagnosis, recognition and categorization of objects, etc.

In order to study with images on computers, some steps are required to digitalize the input images. A digital image may be defined as mathematically a two-dimensional function f(x, y), where 'x' and 'y' represent the spatial coordinates of digital images and the amplitude of 'f' at any pair of coordinates is called the intensity of the image at that point. If the image can be stored in computer memory with finite discrete quantities then it is called a digital image.

In order to get some useful information or conversion from analog space to digital space on images, Image processing techniques which include specific operations are required. These techniques rely on carefully designed algorithms. Performed automatically, are changes that take place in images through these algorithms of image processing. The problem of extracting information from images and interpreting this information has been the incentive for the research community of image processing. Image processing techniques can be used in many fields, including medicine, business, industry, military.

Image processing is improved by interdisciplinary fields with contributions from different arms of science like mathematics and physics, computer and electrical engineering and optics. For the application area, image processing is closely related to the other artificial intelligence fields such as machine learning, human-computer interaction, pattern recognition. The different steps associated with image processing comprise importing the image from an optical scanner or digital camera, analyzing and modifying the image (data compression, image enhancement, and filtering), and creating the desired output image. The general purpose of image processing as a system is the conversion of digital space into a standard pose and obtaining identifiable entity on a uniform background.

Image enhancement can be performed manually or automatically. Manually, the user carried out using Image or special computer software (Photoshop, gimp) may produce finer image pre-processing results. It's preferable to use entirely automated methods to build systems with a large number of images as it is time-consuming to use the manual image processing methods.

In general, the first step in image processing is image acquisition. Image processing accuracy highly depends on this step. A sophisticated camera or modern phone camera can be used to obtain high quality and sharp images. In this step some images distributed from the internet web for training and testing image processing systems. It is also of importance when gathering images to reduce glare and shadows. It is recommended to include in the image some known size reference or scale. Wing images can also be shot by scanning the wings. Without leading to occlusion of the areas of interest, the wings should be well mounted in the images tenanting the largest area possible. Images can be of grayscale or color stored in formats like JPEG, TIFF, or others.

After the image acquisition step, digital image processing is performed. This process consists of two sequential sub-processes: image pre-processing and feature extraction. The First sub-process tries to improve input images and the modified image is its output. This step tries to solve many problems such as noise reduction, occlusion removal, etc. Taking an image as input this step produces a reformed image as an output, which should be suitable for the feature extraction step.

The latter sub-process focuses to extract useful sets of computations either characterizing the whole image or a few components thereof and the output comprises information about the content of the image. The image preprocessing sub-process may perform noise removal, enhancement, and segmentation. In many previous studies, Matlab has been used to facilitate the process of image processing and testing for accurate results before they are implemented in the system or programming language such as Java, C#, Python, etc. In this section, first, we need to process images using Matlab, and we will implement three stages of image processing (Edge Detection (ED), Noise, Graph Modeling (GM)) for a pure and accurate image to be used for matching later.

PART 3

MATERIALS AND METHODS

3.1. MATERIALS

In principle, some of the images were downloaded from the internet to use for the system experience. Each image of an aerial view with a natural background. Since the two sides of the butterfly wing are different, so we will choose the boundary of the butterfly wing to investigate to maintain the consistency. As well to make the image extraction process clear and high resolution, it is best to remove the background.

3.2. METHOD

In this thesis, we will study on feature extraction techniques and CNN architectures in the context of the winged insect (butterflies) classification. The general flowchart is given in Figure 3.1.

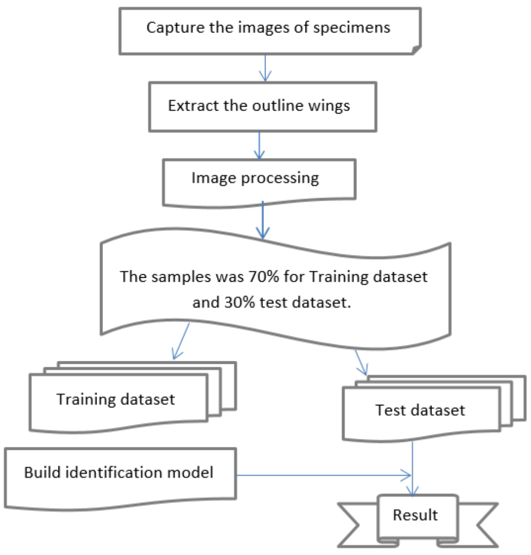

Capture the images of specimens
Extract the outline wings
Image processing
The samples was 70% for Training dataset and 30% test dataset.
Training dataset
Test dataset
Build identification model
Result

Figure 3.1. Flowchart - explains the steps that will be used in this study.

The idea of using (Edge Detection (ED), Noise, Graph Modeling (GM)) for butterfly species illustration is inspired by the reality that butterflies can be seen as a composition of micro-patterns which are consequently expressed by this operator. The method for classifying the images suggested in this study is based on texture descriptors obtained from the butterfly images. As demonstrated in Figure 3.2, into four major steps, is how divided our approach can be:

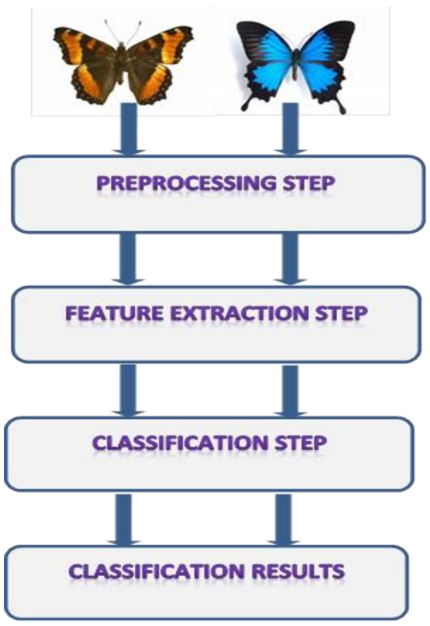


Figure 3.2. Block diagram of the experimental design.

The objective of image preprocessing is cleaning images from various noises. In image processing, noise may occur in the transforming phase of the image from analog to digital form or during the image transmission phase.

In order to perform a useful pre-processing process the key step is to determine the region of objects in the input image. This step is called edge detection. In literature, there are well-known methods for edge detection (Robert’s operator, Prewitt operator, Sobel operator, and Canny edge detector). If the image contains only one object, these operators help to identify to points which are must be a focus. Figure 3.3 shows output images with different edge detection algorithms on a sample butterfly image.

The general approach of edge detection has four steps listed as follows:

1. Smoothing: In this step, noise is detached without spoiling the true edges.
2. Enhancement: To uncover the true edges, filters are applied to the image. This step may be called sharpening.
3. Detection: determine which edge pixels should be gotten rid of as noise and which should be kept (usually, thresholding provides the criterion used for detection).
4. Localization: determine the exact location of an edge (sub-pixel resolution might be required for some applications, that is, estimate the location of an edge to better than the spacing between pixels). Edge thinning and linking are usually required in this step.

Some software (MATLAB) or libraries (OpenCV) has released to using to edge detection. These libraries include many special methods for edge detection (Robert's, Prewitt, Sobel, Canny operators, etc.).

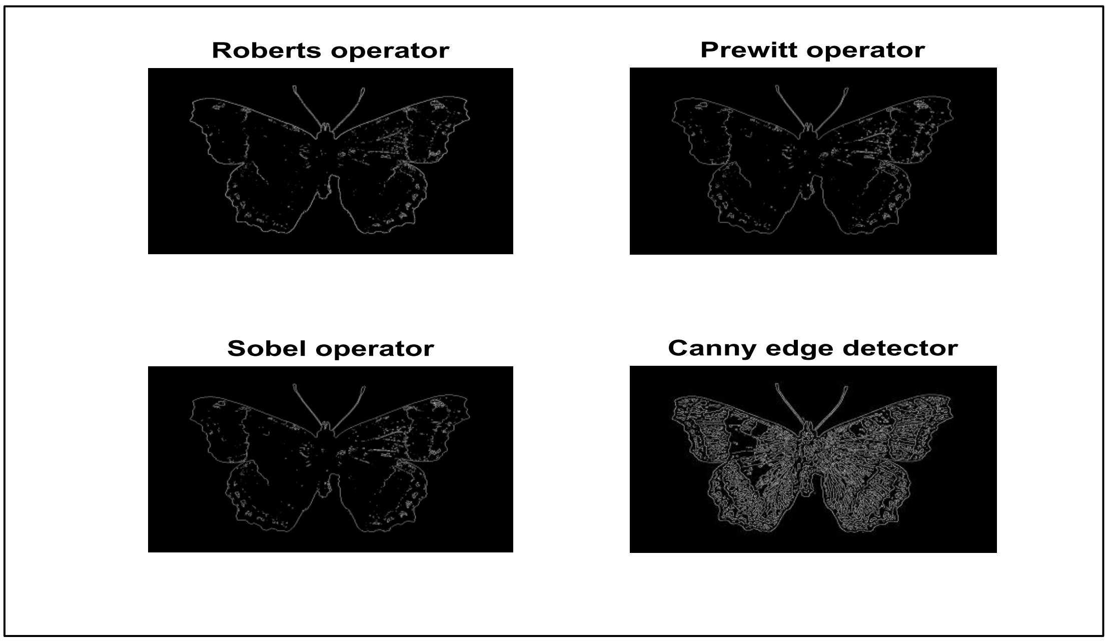


Figure 3.3. Edge detection using various operators.

Robert's operator: The Roberts Cross operator performs on an image a simple, rapid to compute, 2-D spatial gradient measurement. It thus highlights regions of high spatial frequency which often correspond to edges. In its most common usage, the input to the operator is a grayscale image, as is the output. Pixel values at each point in the output represent the estimated absolute magnitude of the spatial gradient of the input image at that point. Horizontal derivative approximate as Equation 3.1 and vertical derivative approximate as Equation 3.2. It's convolution kernel matrices are given in Figure 3.4. The output effect of the Robert Cross operator is shown in Figure 3.5.

$$gx = (z4 - z1) \tag{3.1}$$

$$gy = (z3 - z2) \tag{3.2}$$

$$\underset{Gx}{\begin{array}{|c|c|}\hline +1 & 0 \\ \hline 0 & -1 \\ \hline\end{array}} \qquad \underset{Gy}{\begin{array}{|c|c|}\hline 0 & +1 \\ \hline -1 & 0 \\ \hline\end{array}}$$

Figure 3.4. Roberts Cross convolution kernels.

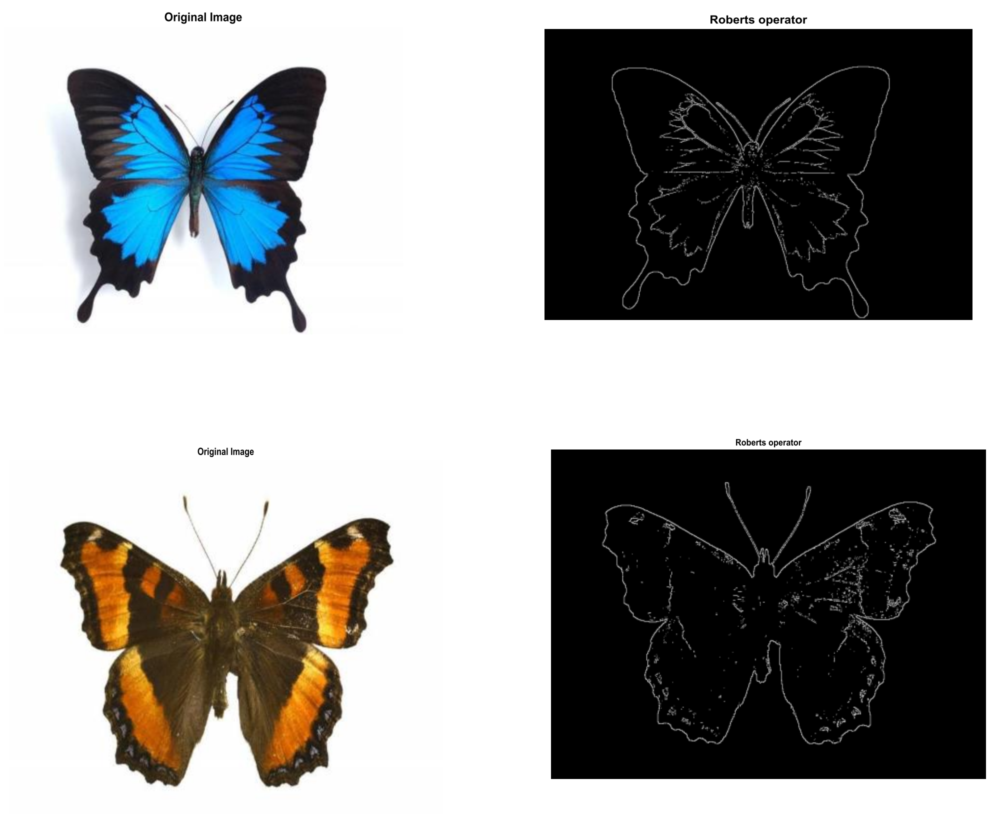
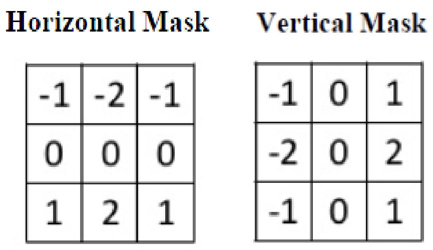


Figure 3.5. Robert's operator of the butterfly image.

Sobel edge detector: When applied to gray-scale images, the Sobel operator calculates the gradient of each pixel's brightness intensity, giving the direction of the larger possible increase from black to white, and also calculates the sum of that direction's shift. A 2-D spatial gradient analysis is performed on an image by the Sobel operator. It is usually used in an input grayscale picture to find the estimated absolute gradient magnitude at each point. The Sobel edge detector uses a pair of 3x3 convolution masks, one estimating the x-direction gradient (columns) and the other estimating the y-direction (rows) gradient. It's very similar to the director in Prewitt. It is also a derivative mask used for the detection of edges. It also measures the horizontal and vertical edge detection. It's got better control of noise. The operator consists of the following pair of kernels of 3x3 convolution (Figure 3.6). The output effect of a Sobel operator is shown in Figure 3.7.



Horizontal Mask

-1	-2	-1
0	0	0
1	2	1

Vertical Mask

-1	0	1
-2	0	2
-1	0	1

Figure 3.6. Masks for Sobel edge detection operator.

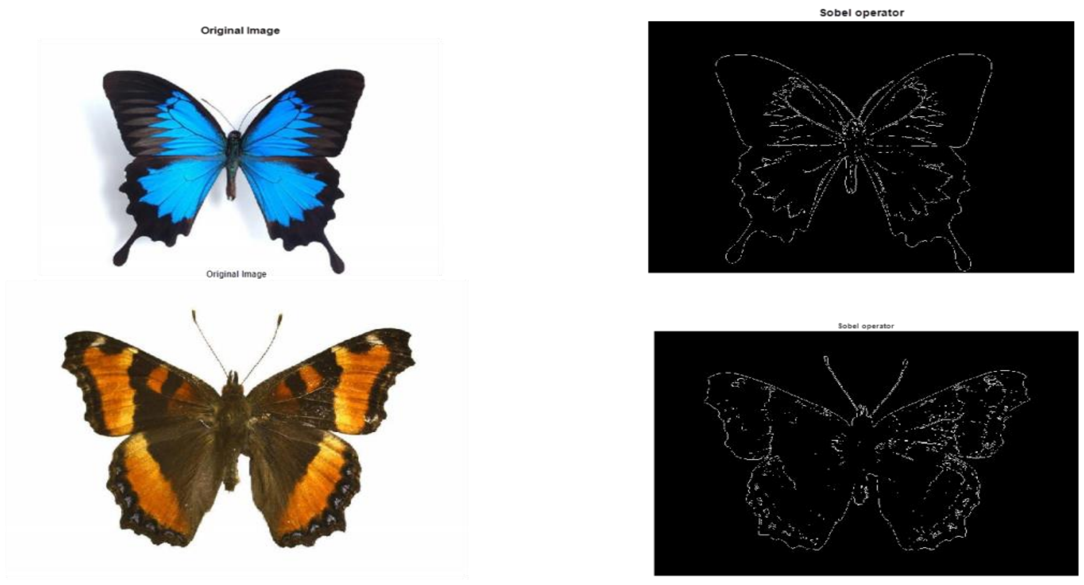
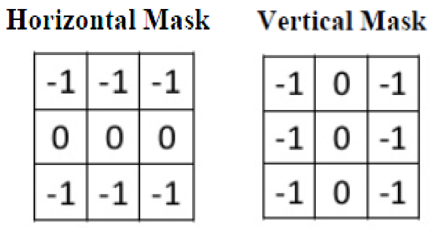


Figure 3.7. Sobel edge detector of butterfly image.

Prewitt edge detector: It is used for edge detection than detect two types of edge horizontal or vertical. The edges are computed by using the difference between corresponding pixel intensities of an image. The masks used for edge detection are all known derivative masks and this operator is called the derivative operator. The kernel matrices are given in Figure 3.8. Also, the output image of this operator is given in Figure 3.9.



Horizontal Mask

-1	-1	-1
0	0	0
-1	-1	-1

Vertical Mask

-1	0	-1
-1	0	-1
-1	0	-1

Figure 3.8. Masks for Prewitt edge detection operator.

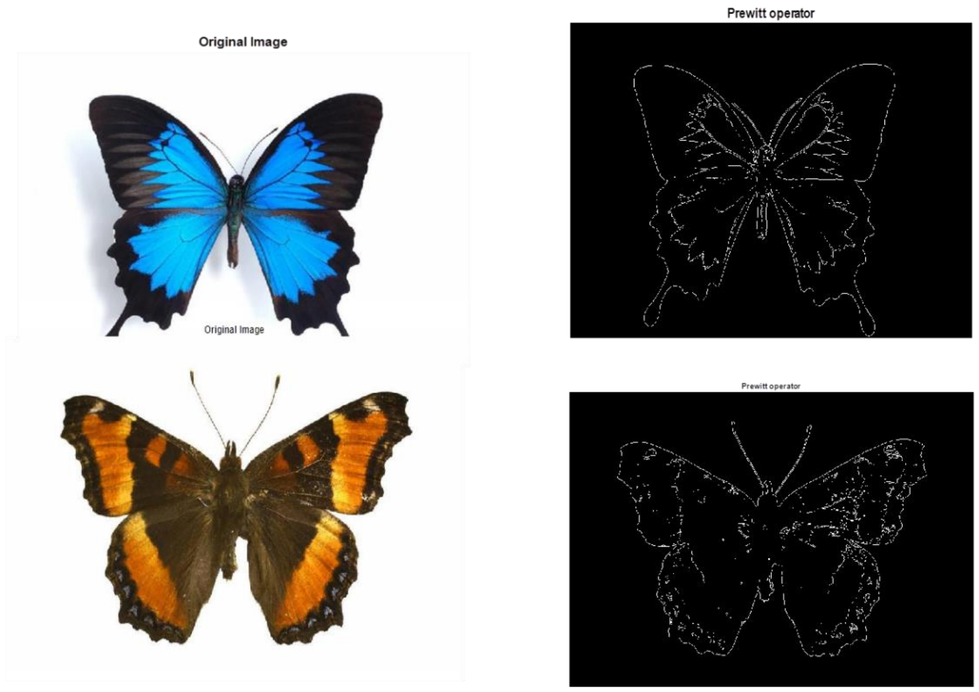


Figure 3.9. Prewitt Operator of the butterfly image.

Canny Edge Detector**:** The system of Canny seeks edges by looking for local gradient maxima. The gradient will be determined using a Gaussian filter derivative. The method uses two thresholds to detect strong and weak edges and includes weak edges only when related to strong edges in the performance. Therefore, this approach can rarely be fooled by noise and more liked than the others. The output image of this operator is given in Figure 3.10.

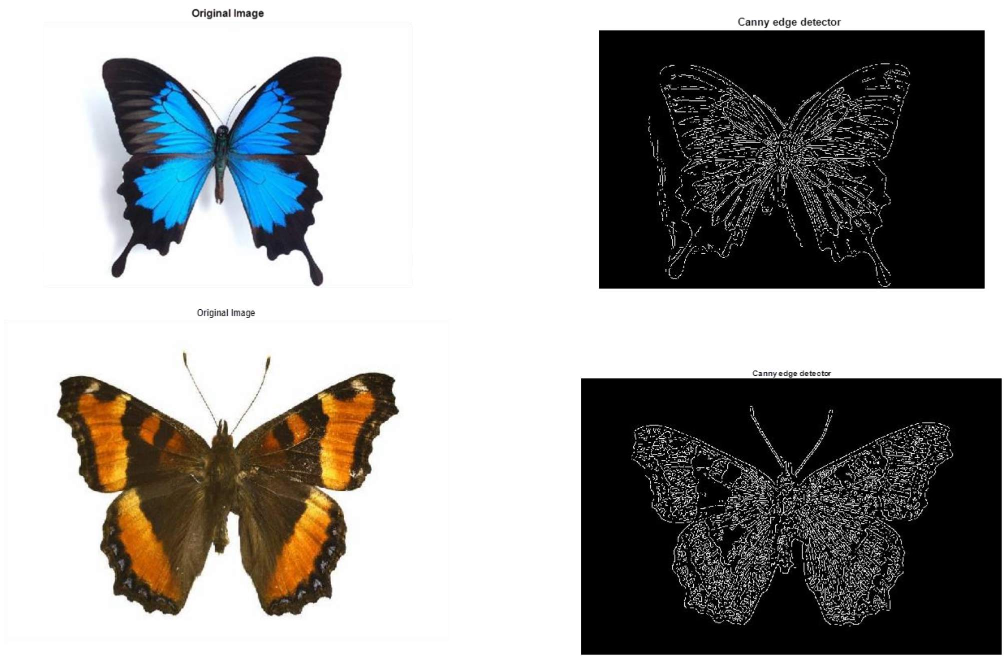


Figure 3.10. Canny edge detector of the butterfly image.

There are various types of noise (speckle, Gaussian, salt-and-pepper). Such forms of noise can be used as special effects in the picture using a butterfly image as shown in Figure 3.11. In this step, RGB-to-gray is first converted and different types of noise are applied to the image through the software.

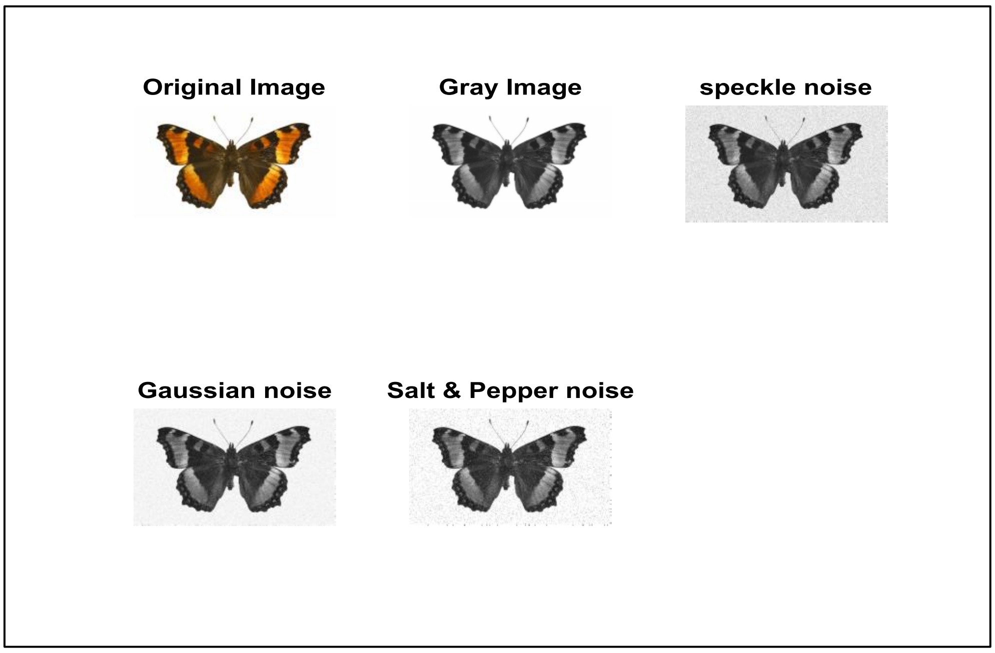


Figure 3.11. Special effects in an image using different types of noise.

Salt and pepper noise: Due to sharp and sudden changes in the image signal, this noise comes about in the image as shown in Figure 3.12. For images that are distorted by salt and pepper noise, the maximum and minimum values in the dynamic range are taken by the noisy pixels. In general, salt and pepper noise is caused by malfunction of pixel elements in-camera sensors, faulty memory locations or timing errors in the digitization process.

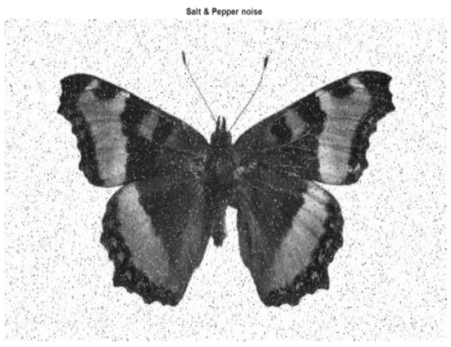


Figure 3.12. Salt and pepper noise of butterfly image.

Gaussian noise: is evenly distributed over the signal. This infers that the sum of the true pixel value and a random Gaussian distributed noise value are each pixel in the noisy image. At every level, the noise is independent of the pixel value size. White

Gaussian noise is a unique case, with the distribution of values in a similar way and at any pair of time-independent statistically. White noise inherits its name from white light. During the processing, for example, sensor noise brought about by poor lighting or high temperature or transmission, the main sources of Gaussian noise in digital images emerge. Figure 3.13 is shown an example of Gaussian noise production on the butterfly image.

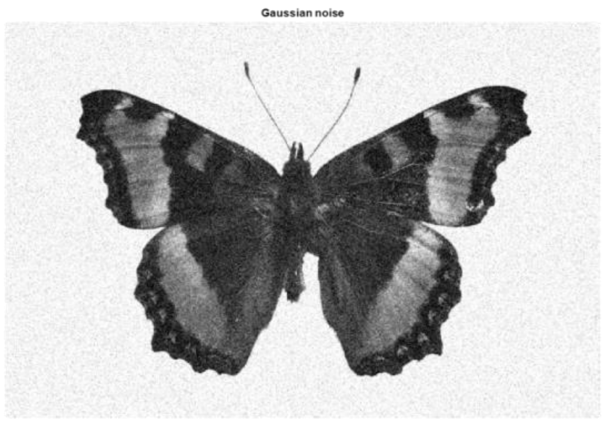


Figure 3.13. Gaussian noise of butterfly image.

Speckle noise: in contrast to the Gaussian and salt pepper noise, speckle noise is multiplicative kind of noise. By multiplying a random value with picture pixel values this noise can be cast and represented as

$$P = I + n * I \tag{3.3}$$

P standing for speckle noise distribution image, I stand for the input image and n for the uniform noise image by means o and variance v. The output image of the speckle noised butterfly image is given below in Figure 3.14.

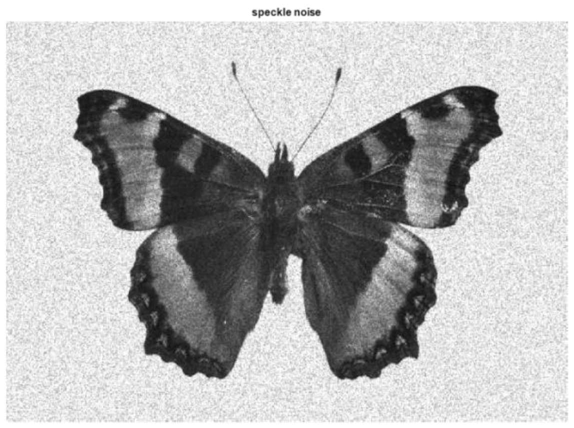


Figure 3.14. Speckle noise of butterfly image.

An image histogram provides a comprehensive description of an image. It represents the occurrence of different levels of gray relative to the frequencies. Like other histograms, the histogram of an image often displays frequency. But the frequency of pixel intensity values is shown by an image histogram. The equalization of the histogram is used to improve contrast. Contrast is not always going to be an increase. There may be some cases where the equalization of the histogram may be worse. The contrast is diminished in those situations. Figure 3.15 shows us plotting the histogram of the original image and the histogram-equalized image, using MATLAB commands we can use these types of butterfly image:

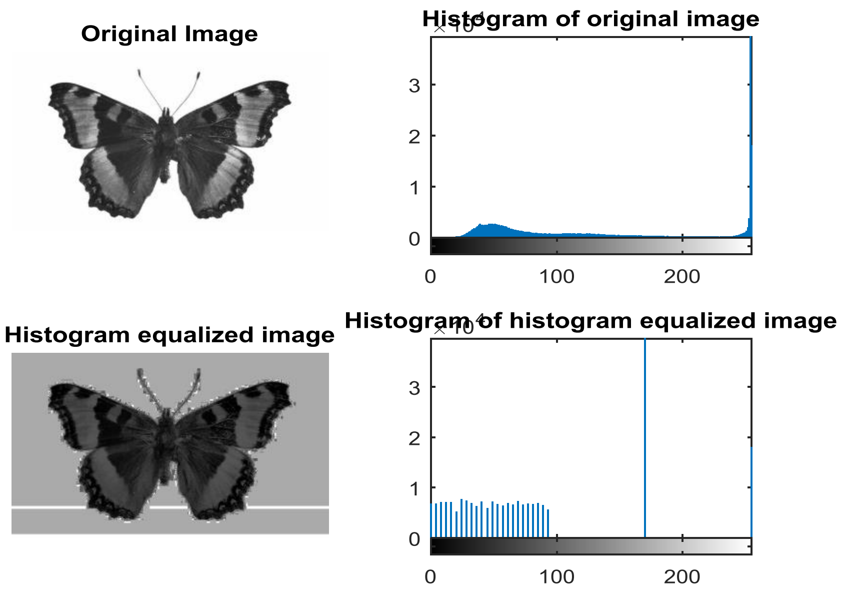


Figure 3.15. Histogram modeling.

Histogram

There exist many applications of histograms in image processing. The first is the image's analysis. By looking at its histogram, we can predict an image. It's like searching for a piece of a body's x-ray. The other use of the histogram is for purposes of brightness. The histograms have broad image brightness applications. Not only in brightness but also in histograms are used to change a picture's contrast. The equalization of an image is another significant use of histograms. And last but not least, thresholding is widely used by the histogram. This is mostly used in the view of the machine.

Extracting and selecting features

Four characteristics of shape, texture, light, and vein are used in the conventional identification theory to distinguish different butterfly species. According to several studies, due to the colorful scales on the wings, it was discovered that the vein features were tough to acquire from the digital image of the butterflies. In fact, butterfly specimens would have different colors for the same species due to the different preservation period. But after the full metamorphosis, the form and texture of the butterfly wings are relatively stable. And the shape of different samples of similar species is less variable than the texture. In addition, there is a significant difference between different species in the shape of butterfly wings. The shape characteristics are therefore used for the identification of the basic level and the texture characteristics are used for further identification. There are many feature selection techniques that are now being used widely in these days, such as Convolutional Neural Networks (CNNs), Independent Component Analysis (ICA), Principal Component Analysis (PCA), and Linear Discriminant Analysis (LDA), this analysis will concentrate on Convolutional Neural Networks (CNN).

PART 4

ARTIFICIAL NEURAL NETWORK

Artificial neural network (ANN), which is a machine learning method, is a parallel computing system to mapping function from inputs to outputs. Basically, ANN includes mathematical nonlinear methods originated to perform accomplish the complex exercise. ANN is mimicked biological brain networks having the capability to the adaptation to nature, learning and recall of information. Over the years, ANN's being used in numerous fields of machine learning applications comprising image processing, identification, classification, optimization, power systems, signal processing, and control system.

Figure 4.1. shows the general overview of ANN which includes three types of layers namely, input, hidden and output. The input layer brings the initial data (input features or decision variables) into the system for further processing by subsequent layers of artificial neurons. The hidden layer is connected to the input and output layers. Each connection in this layer takes weight and produces output through an activation function. The output layer produces outputs of ANN.

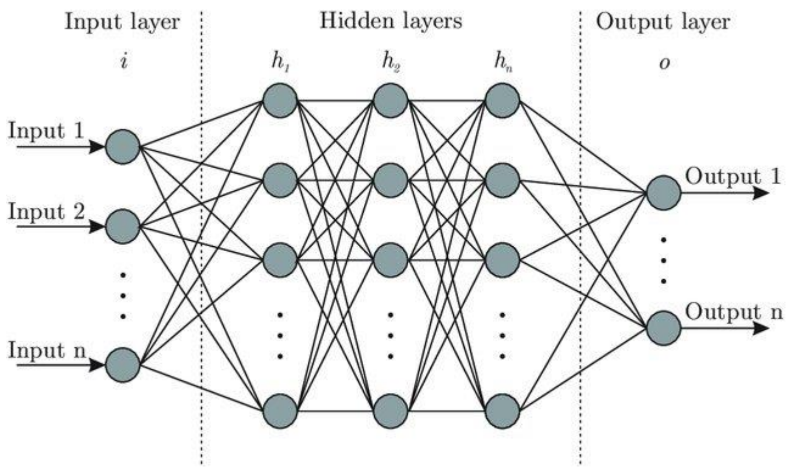


Figure 4.1. A general overview of ANN.

An ANN is generally formed from multiple connected processing nodes called as artificial neurons and their connection between them. The network consists of seven major components:

1. Weighting Factors
2. Summation Function
3. Transfer Function
4. Scaling and Limiting
5. Output Function
6. Error Function
7. Learning Function.

A classical feed-forward multi-layer perceptron (MLP) architecture can be used for regression or classification tasks. It has two phases namely training and test. Generally, training and test phases use the same dataset which is divided into two groups. In the training phase, for a predetermined specific dataset, until the ANN reaches the desired accuracy, neuron's wights are upgraded. The accuracy of the architecture is proportional to the selected parameters and dataset. Thus, parameter optimization of ANN and the fine-grained datasets is essential. If the dataset has no enough samples to cover all its aspects, the accuracy of the network will reduce.

The training phase performs two ways supervised (labeled data) or unsupervised (unlabelled data). For supervised training of MLPs, one of the very general methods is the back-propagation algorithm. This algorithm is grounded on the gradient descent. The weights of the network are randomly initialized and then changed in a direction along the negative gradient of the mean squared error (MSE) so as to minimize the difference between the network output and the desired output.

The training of the network comprises four steps: (1) determining the training data, (2) design the ANN, (3) train the ANN lastly (4) determine the accuracy of the network outputs with the test dataset. After a determined step of the training process, the ANN will have an adequate ability to perform a nonlinear pattern association between input and target. A well-trained network will effortlessly anticipate the output when a new input data is applied to it.

4.1. ANN CHARACTERISTICS

ANN Characteristics Mainly computers are good in calculations that mostly take input process and after which it provides the result based on calculations that are performed on different algorithms that are programmed in the application whilst ANN improves their own commands, the more decisions they make, the better decisions can be made. The network characteristics of ANN is given in Figure 4.2.

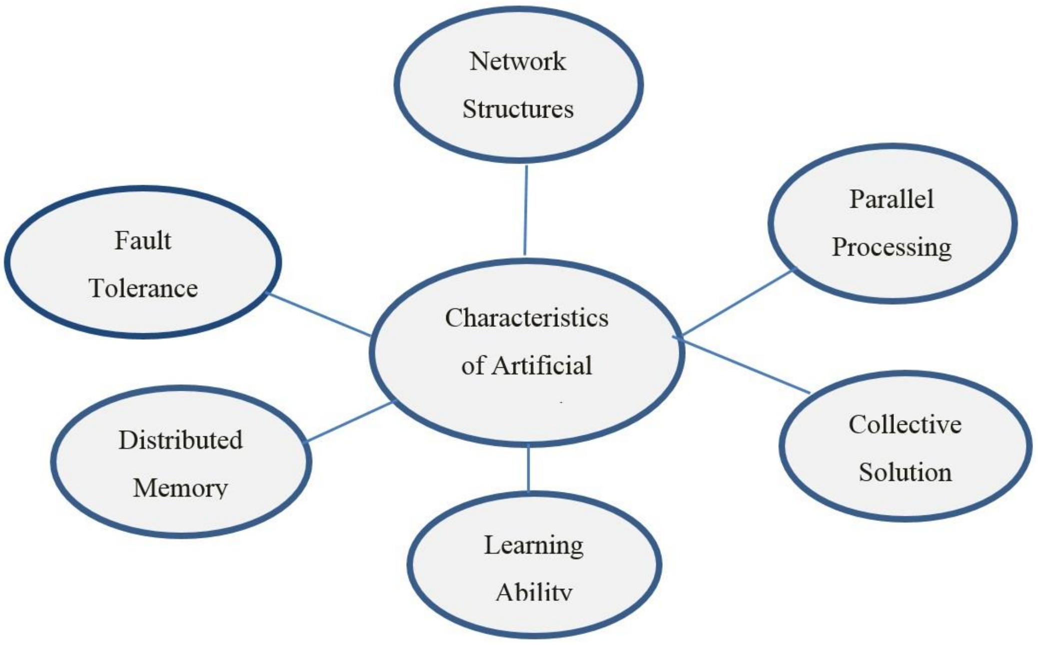


Figure 4.2. Artificial neural network characteristics.

4.2. NETWORK STRUCTURE

ANN's network configuration should be uncomplicated and straightforward. There exist primarily two types of recurring and non-recurring structures. Also known as the Auto Associative or Feedback Network is the Recurrent Structure and the Non-Recurrent Structure is also known as the Associative or Feed-forward Network. The signal travels in one direction only in the feed-forward network, and as for Feedback

Network, the signal travels both ways by adding loops into the network. Below are the particulars showing the way signals are fed-forward and feedback in both the network structures.

4.3. IMAGE DATA PRE-PROCESSING FOR NEURAL NETWORKS

In the past few years, deep learning has really entered the mainstream. Deep learning uses neural networks with many hidden layers (dozens in the state of the art of today) and requires large amounts of data from training. In perceptual tasks such as vision, voice, language processing, these models have been particularly effective in gaining insight and achieving human-level accuracy. Several decades ago, theoretical and mathematical foundations were laid. Two factors have led primarily to the growth of machine learning are

1. The availability of massive data sets/training examples in multiple domains and
2. Advancements in raw computing power and the rise of effective parallel hardware.

The most common parameters of image data input are the number of images, the height of the image, the width of the image, the number of channels and the number of pixel rates.

PART 5

LITERATURE REVIEW

There are many studies in the literature related to insect classification and identification. In the last two decades, there has been a significant increase in the number of researches on neural networks due to the availability of fast computers and a large amount of data. Keanu Buschbacher et al. proposed a new approach called DeepABIS which is based on the Automated Bee Identification System (ABIS). ABIS that is a fully automated approach aims to recognize bee species from the pattern on their forewings. DeepABIS has three important features: (1) DeepABIS employs automated feature generation to reduce the efforts of training the system using convolutional neural networks (CNNs), (2) DeepABIS enables web portals and participatory sensing using mobile smartphones and a cloud-based platform for data collection, (3) DeepABIS is adaptable to other insects beyond bees, such as butterflies, flies, etc. DeepABIS achieved to identify wild bees on the ABIS dataset an average top-1 accuracy of 93.95% and a top-5 accuracy of 99.61%. They then adapted DeepABIS to a butterfly dataset including 5 classes and obtained classification results with an average top-1 accuracy of 96.72% and a top-5 accuracy of 99.99% [16].

Linan Feng et al. develops an automated moth species identification and retrieval system using computer vision image processing techniques. The system is a probabilistic model that infers attributes from visual features of moth images in the training set and learns the co-occurrence relationships of the attributes [17]. They tested their system on a dataset consisting of 4530 images of 50 moth species and estimated the best recognition accuracy of 70% for some species. The lowest accuracy they obtained is approximately 34% for species Narcus Burns. The main difference between their work and the other similar works for insect classification is that they provide intermediate-level features which are Semantically Related Visual

(SRV) attributes on moth wings such as eyespots, white central band, marginal cuticle, and snowflake mosaic.

Kaya et al. [3] proposed two different local binary patterns (LBP) descriptors for detecting special textures in images. The first one is based on the relations between the sequential neighbors of a center pixel with a specific distance and the other one is relies on determining the neighbors in the same orientation through the central pixel parameter. They tested their descriptors to identify butterfly species on lab-based images of 140 butterflies collected in Van city of Turkey. The highest accuracy they obtained for classifying butterflies by the artificial neural network is 95.71%. Kaya and Kaycı [4] developed a model for butterfly identification based on color and texture features by the artificial neural network using the gray level co-occurrence matrix with different angles and distances. The accuracy of their method reached 92.85%.

Wen and Guyer [5] developed a model which combines local and global features for insect classification via five well-known classifiers that are K nearest neighbor classifier (KNNC), normal densities based linear classifier (NDLC), minimum least-square linear classifier (MLSLC), nearest mean classifier (NMC) and decision tree (DT). Their experimental results tested on images collected from actual field trapping for training yielded the classification rate of 86.6%.

Xie et al. [6] improved a learning model for the classification of insect images using advanced multiple-task sparse representation and multiple-kernel learning techniques. They tested the proposed model on 24 common pest species of field crops and compared it with some recent methods.

Feng et al. [7] improved an insect species identification and retrieval system based on wing attributes of moth images. Their retrieval system is based on CBIR architecture which is not about having a final answer but just providing a match-list to the user. The final decision can be made by an expert.

Yao et al. [8] designed a model to automate rice pest identification using 156 features of pests. The authors tested the proposed model on a few species of Lepidoptera rice pests with middle size using the support vector machine classifier. Another study for pest detection and recognition was carried out by Faithpraise et al. using the k-means clustering algorithm and correspondence filters [9].

Leow et al. [10] developed an automated identification system of copepod specimens by extracting morphological features and using ANN. They used seven copepod features of 240 sample images and estimated an overall accuracy of 93.13%.

Zhu and Zhang [11] proposed a method to classify lepidopteran insect images by integrated region matching and dual-tree complex wavelet transform. They tested their method on a database consisting of 100 lepidopteran insects of 18 families and estimated the recognition accuracy of 84.47%.

Mayo and Watson [12] showed that data mining techniques could be effective for the recognition of insect species. They used WEKA which is a data mining tool with different classifiers such as Naïve Bayes, instance-based learning, random forests, decision trees, and support vector machines. WEKA was able to achieve an accuracy of 85% using support vector machines to classify live moths by species. Another work for identification of live moths was conducted by Watson et al. [13] who proposed an automated identification tool named DAISY by analyzing wing attributes and shapes. However, DAISY requires user interaction for image capture and segmentation.

Silva et al. [14] aimed to investigate the best combination of feature selection techniques and classifiers to identify honeybee subspecies. They found the best pair as the combination of Naïve Bayes classifier and the Correlation-Based feature selector in their experimental results among seven combinations of feature selectors and classifiers.

Wang et al. [15] designed an identification system of insect images at the order level. They extracted seven features from insect images. However, the authors manually removed some attached to insects such as pins. Their method has been tested on 225 specimen images from 9 orders and sub-orders using an artificial neural network and 7 features with an accuracy of 93%. They achieved an accuracy of 92% using a support vector machine and 7 features.

Angelo et al. [18] created a mobile application for the Android platform for insect identification using the Inceptionv3 model, which is a convolutional neural network architecture. The Android application, named Insectify, provides offline functionalities to general users. 10 insects are considered in their study. These are butterfly, bee, beetle, cockroach, cicada, house fly, dragonfly, mantis, wasp, and grasshopper. They conducted the tests to assess the effectivity of the model on images with varying color mode, orientation, and background. The model achieved a maximum accuracy of 95.33% based on the model's first guess of the correct insect common name. The model was able to isolate distinct features in an image captured by mobile phones. Therefore, Insectify may be a useful and portable tool to provide insect identification beneficial to different fields of studies and localities such as farmlands, forests and common households.

Xue et al. proposed a new model to identify butterfly species [19]. Their model includes two significant improvements: extracting the gray-level co-occurrence matrix (GLCM) features in image blocks and weight-based k-nearest neighbor (KNN). The model they improved extracts GLCM features from three image blocks of butterfly images. They do not work on the whole butterfly area. Because extracting GLCM features on the whole area will increase the running time and some important features in the local area will be ignored. After computing GLCM features, they used a weight-based KNN search algorithm for classification. The identification accuracy of their method for 10 butterfly species reached 98%.

PART 6

METHODOLOGY

In this thesis, our study consists of two stages. In the first stage, we studied on lab-based butterfly images taken on under a fixed protocol. The species of butterflies in these images are identified by expert entomologists. We used a total of 140 images taken in [20] for lab-based butterfly images of 10 species. After applying some preprocess to the images such as histogram equalization and background removing, we extracted several features from the butterfly images. Finally, we used an artificial neural network in MATLAB version R2014b using the Neural Network Toolbox for butterfly identification. The ANN model could achieve an accuracy of 96.5% using one input layer, one hidden layer, and one output layer. The details for the model used and the other information are given in the following sections.

In the second stage of the thesis, we studied on field-based butterfly images. Field-based images of butterflies are taken in nature without any protocol or constraints. These pictures are generally taken using either a mobile device or a digital camera. We gathered 44659 images of 104 species from the website of Butterflies Monitoring & Photography Society of Turkey that consists of field-based butterfly images [20]. To identify the species of the field-based butterfly images, Convolutional Neural Networks which is a class of deep learning was used. Each class was divided into two parts automatically: the training part (80%) and the testing part (20%). Approximately 80% of success was achieved for both test and training data using CNN architecture.

The general flowchart of these two stages is given in Figure 6.1.

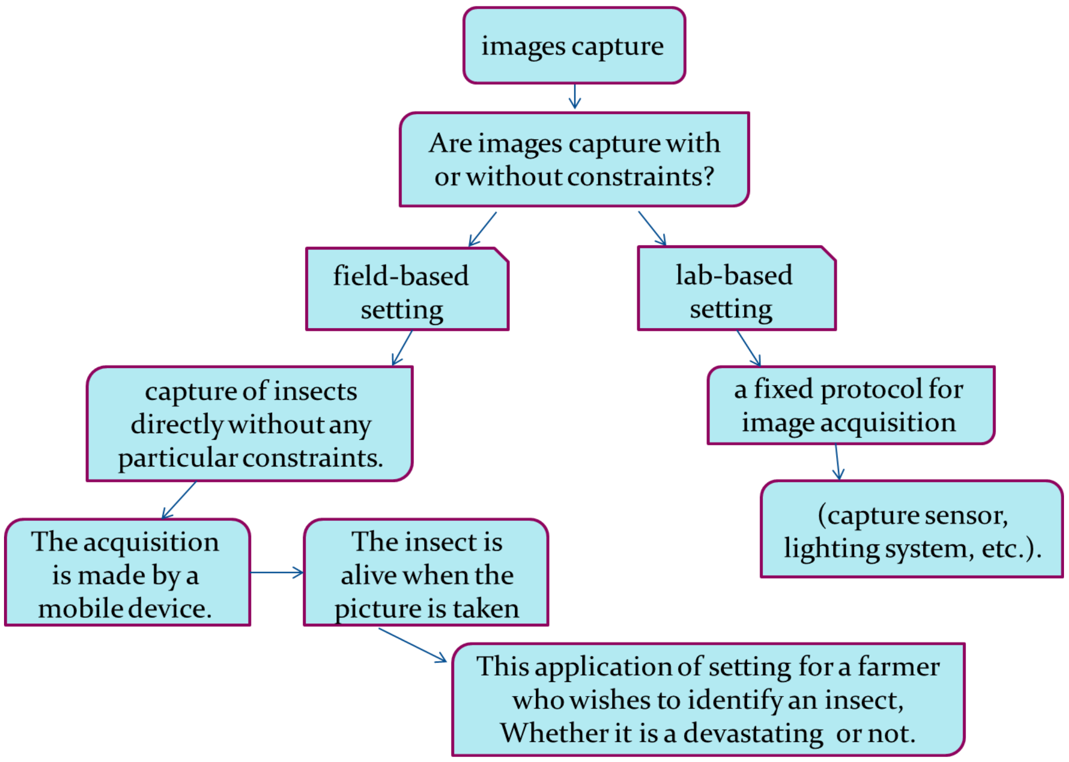

images capture
Are images capture with or without constraints?
field-based setting
lab-based setting
capture of insects directly without any particular constraints.
a fixed protocol for image acquisition
(capture sensor, lighting system, etc.).
The acquisition is made by a mobile device.
The insect is alive when the picture is taken
This application of setting for a farmer who wishes to identify an insect, Whether it is a devastating or not.

Figure 6.1. Flowchart to explain the research methodology.

6.1. ADVANTAGES AND DISADVANTAGES

According to the common use of mobile devices, field capture is very easy. A lot of users can take butterfly photos in nature. These photos do not require any special hardware and a fixed protocol such as the position of butterflies, the shooting angle, butterfly distance, occlusion, lighting, and background complexity. Therefore, researchers can collect many sample images by themselves or from the internet to construct a dataset of butterflies. However, identification systems working on field-based images are difficult to design and do not generally yield good accuracy.

Collecting the lab-based butterfly dataset is difficult, because it needs a fixed protocol, identifying them is time-consuming work for entomologists, and the number of butterflies included in the butterfly dataset is not comprehensive. However, preprocessing operations such as background removing is quite easy. Therefore, machine learning techniques for classification give more accurate results.

6.2. FOREGROUND / BACKGROUND DETECTION

Before related features are extracted, the system needs to know which region of the image is important. This part is taken care of by segmentation. In some cases, segmentation is learned in a supervised manner as in [60, 61]. The study in [61] feeds the whole image to the classifier, which is supposed to learn to discriminate only on the foreground and therefore to recognize it. The approach in [60] is based on learning on both negative and positive sample images. Negative sample images are images where there is no any insect but only foreground. Learning on such images enables the system to ignore the background. When segmentation is irrelevant, it may be asked to the user. The user can select the region of interest on the given interface by drawing its outlines. However, manual segmentation might be a tedious and time-consuming task considering how numerous the images can get. That is why segmentation is worth automated most of the time.

Some segmentation techniques are based on thresholding which is basically splitting the image histogram into several groups that correspond respectively to the object and the background. The simplest way of performing thresholding is to set the intensity value that separates these two groups. The intensity value can be set statically in the program or by the user who can select the one which gives the best result as in [22,23]. Another way of performing thresholding is to see it as a clustering problem where two or more clusters (which are the regions) have to be formed [24-27].

Otsu's method criterion is about choosing the clusters such that the intra-cluster variation is minimized while the inter-cluster one is maximized [10,25,27]. k-means is used onto the color space to search for centroids representing the different regions in the image based on color similarity [9,26]. The study in [9] uses ISODATA, a clustering algorithm that builds clusters with a given standard deviation threshold. The studies in [10,27] use mean shift clustering in the color space as a preprocessing step to Otsu.

There are other techniques to segment objects from the images. The papers in [28,29] use active contours (snake) that take a simple thresholding mask as a seed point to get a more accurate segmentation. The work in [7] uses background segmentation (which implies the background to be constant). The study in [30] supposed the object in the image gives the longest outline while the research in [6] searches for lines on wing images and uses the symmetry of Lepidoptera as a detection criterion.

6.3. MATERIALS

6.3.1. Dataset

The number of butterfly species in the world varies from 15000 to 21000 according to a recent estimate in [1]. In this thesis, we studied two types of images, field-based images, and lab-based images.

6.3.1.1. Field-Based Images

We gathered 44659 images in different 104 classes from the website of Butterflies Monitoring & Photography Society of Turkey [20]. When images were downloaded from the website, we labeled the species which were approved by professionals. The dataset was divided into 104 different classes using these information labels. Since many classes have a few image samples, we restricted it to include 10 classes with the most sample size as shown in Table 6.1. The new dataset has 17.769 images. The number of classes has been reduced to 10 in order to increase the classification accuracy and to remove the classes with fewer samples. Since all images on the website are of different resolutions, they were resized to 224x224 pixels. Sample images for each class are given in Figure 6.2. Since the input images were captured from wide field-of-view cameras, they have some characteristics such as occlusion and background complexity.

Table 6.1. Name and the number of images per butterfly's genus in the dataset.

Genus	**Images**
Polyommatus	5559
Lycaena	2472
Melitaea	2235
Pieris	1530
Aricia	1080
Pyrgus	1036
Plebejus	1003
Nymphalis	989
Melanargia	943
Coenonympha	922

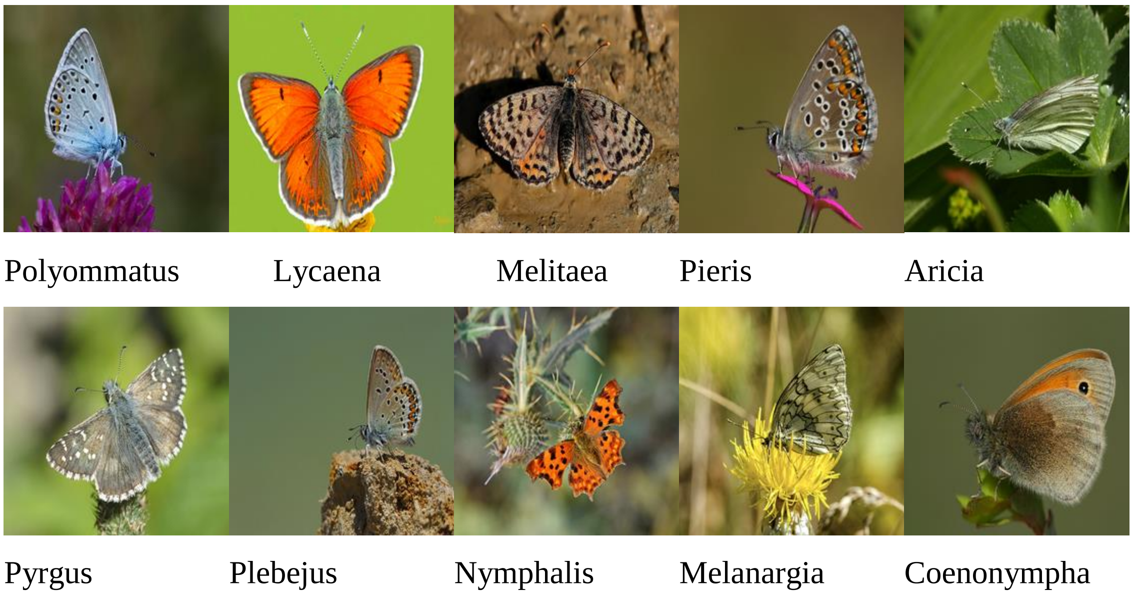


Figure 6.2. Sample images in the field-based dataset.

6.3.1.2. Lab-Based Images

We used images of 140 specimens from 14 butterfly species (see Figure 6.3). For each species, there were 14 specimens.

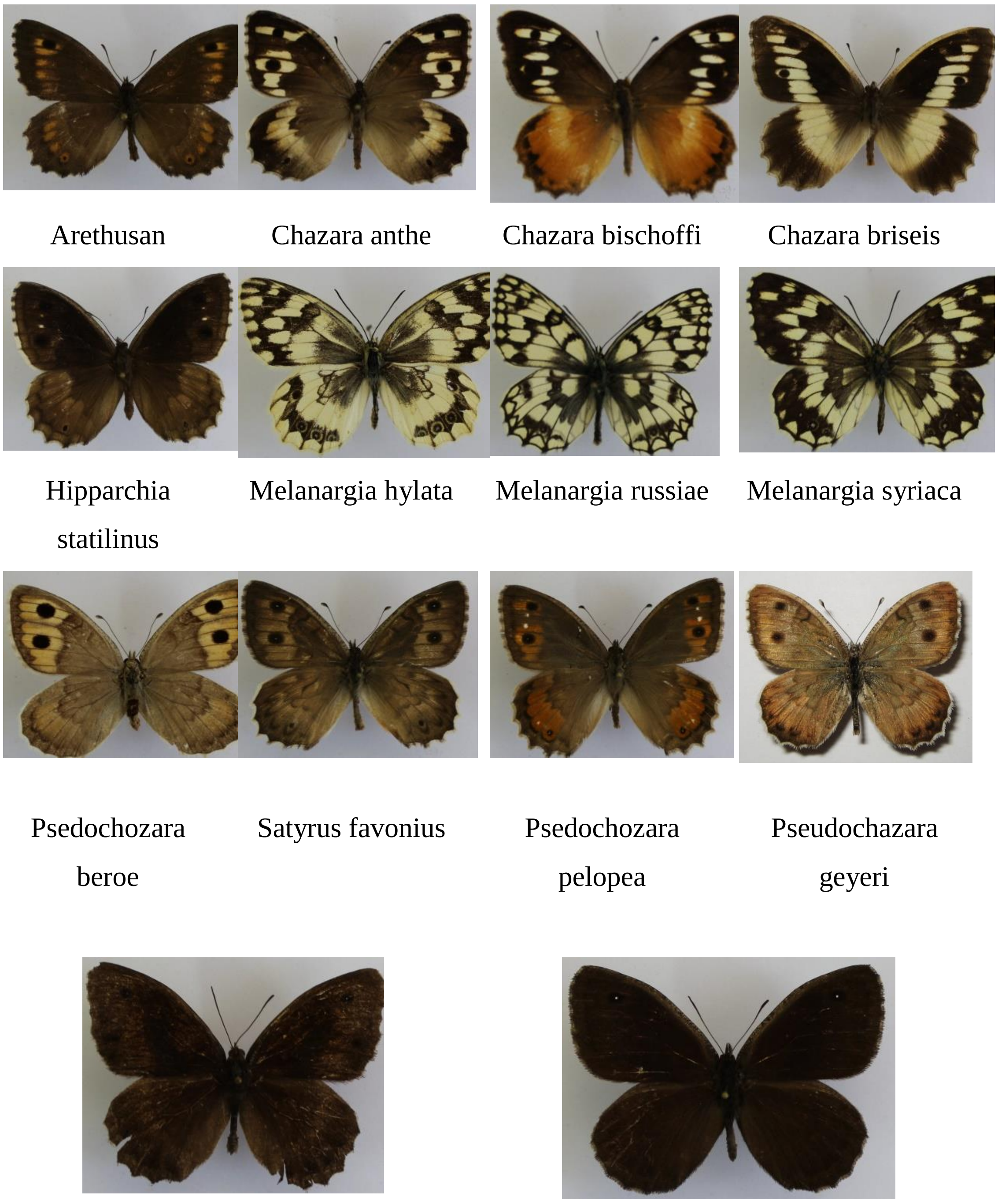


Figure 6.3. Lab based butterfly images with 14 butterfly species.

6.3.2. Image Preprocessing Step

The preprocessing procedure to segment the butterfly from the input image consists of seven steps. They are shown in Figure 6.4.

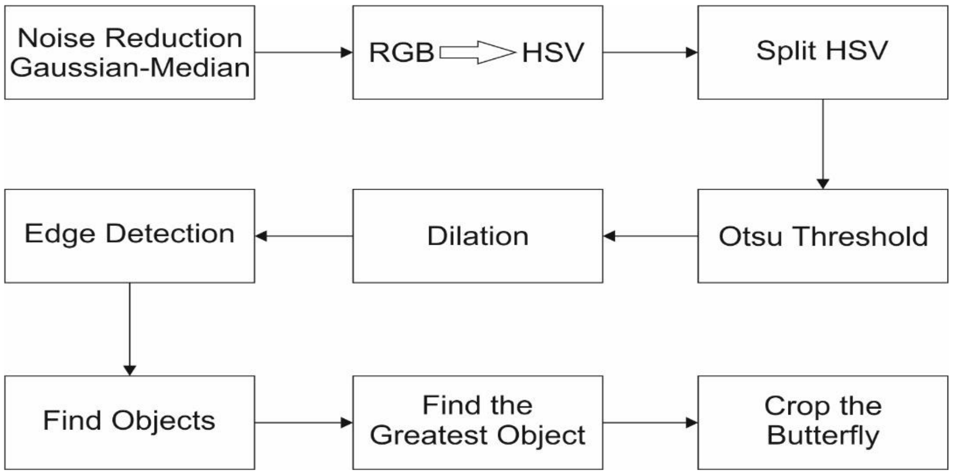
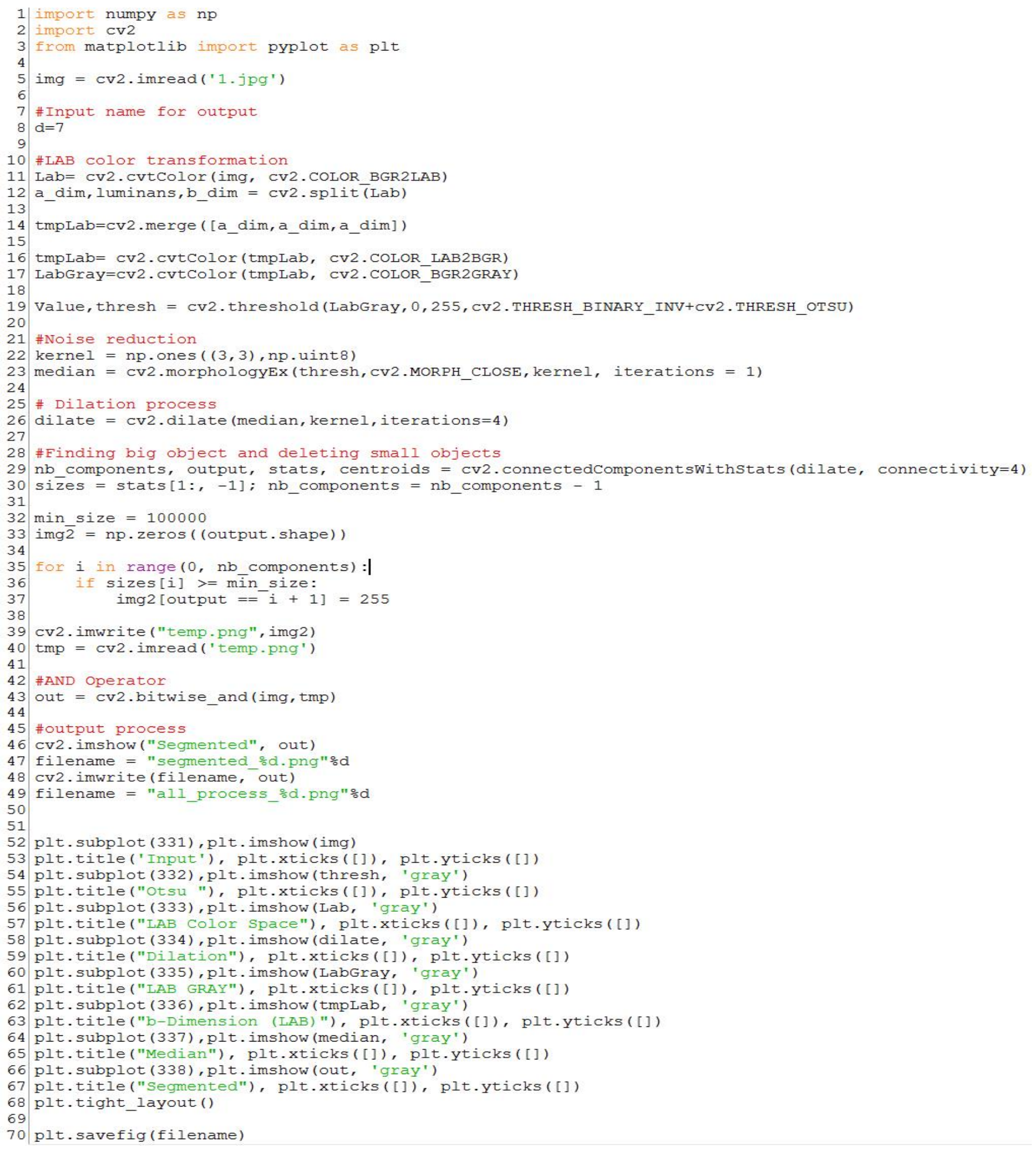


Figure 6.4. Image preprocessing steps.

6.3.3. Implementation

We implemented image preprocessing steps on Python according to Figure 6.4. The python script is shown in Figure 6.5.



```python
 1 import numpy as np
 2 import cv2
 3 from matplotlib import pyplot as plt
 4
 5 img = cv2.imread('1.jpg')
 6
 7 #Input name for output
 8 d=7
 9
10 #LAB color transformation
11 Lab= cv2.cvtColor(img, cv2.COLOR_BGR2LAB)
12 a_dim,luminans,b_dim = cv2.split(Lab)
13
14 tmpLab=cv2.merge([a_dim,a_dim,a_dim])
15
16 tmpLab= cv2.cvtColor(tmpLab, cv2.COLOR_LAB2BGR)
17 LabGray=cv2.cvtColor(tmpLab, cv2.COLOR_BGR2GRAY)
18
19 Value,thresh = cv2.threshold(LabGray,0,255,cv2.THRESH_BINARY_INV+cv2.THRESH_OTSU)
20
21 #Noise reduction
22 kernel = np.ones((3,3),np.uint8)
23 median = cv2.morphologyEx(thresh,cv2.MORPH_CLOSE,kernel, iterations = 1)
24
25 # Dilation process
26 dilate = cv2.dilate(median,kernel,iterations=4)
27
28 #Finding big object and deleting small objects
29 nb_components, output, stats, centroids = cv2.connectedComponentsWithStats(dilate, connectivity=4)
30 sizes = stats[1:, -1]; nb_components = nb_components - 1
31
32 min_size = 100000
33 img2 = np.zeros((output.shape))
34
35 for i in range(0, nb_components):
36     if sizes[i] >= min_size:
37         img2[output == i + 1] = 255
38
39 cv2.imwrite("temp.png",img2)
40 tmp = cv2.imread('temp.png')
41
42 #AND Operator
43 out = cv2.bitwise_and(img,tmp)
44
45 #output process
46 cv2.imshow("Segmented", out)
47 filename = "segmented_%d.png"%d
48 cv2.imwrite(filename, out)
49 filename = "all_process_%d.png"%d
50
51
52 plt.subplot(331),plt.imshow(img)
53 plt.title('Input'), plt.xticks([]), plt.yticks([])
54 plt.subplot(332),plt.imshow(thresh, 'gray')
55 plt.title("Otsu "), plt.xticks([]), plt.yticks([])
56 plt.subplot(333),plt.imshow(Lab, 'gray')
57 plt.title("LAB Color Space"), plt.xticks([]), plt.yticks([])
58 plt.subplot(334),plt.imshow(dilate, 'gray')
59 plt.title("Dilation"), plt.xticks([]), plt.yticks([])
60 plt.subplot(335),plt.imshow(LabGray, 'gray')
61 plt.title("LAB GRAY"), plt.xticks([]), plt.yticks([])
62 plt.subplot(336),plt.imshow(tmpLab, 'gray')
63 plt.title("b-Dimension (LAB)"), plt.xticks([]), plt.yticks([])
64 plt.subplot(337),plt.imshow(median, 'gray')
65 plt.title("Median"), plt.xticks([]), plt.yticks([])
66 plt.subplot(338),plt.imshow(out, 'gray')
67 plt.title("Segmented"), plt.xticks([]), plt.yticks([])
68 plt.tight_layout()
69
70 plt.savefig(filename)
```

Figure 6.5. Python script of the preprocessing step.

6.3.4. Explanation of The Script

In line 1, we import numpy package for scientific computing

In line 2, we import OpenCv package for image processing

In line 3, we import Pyplot module. Pyplot is a matplotlib module which provides a MATLAB-like interface to plot figures.

In line 5, we read the input butterfly image as img.

In line 11, The cvtColor function in OpenCv is the function that performs transitions between color spaces. The COLOR_BGR2LAB parameter is used to convert from BGR to LAB (CIE). As another parameter, the cvtColor function retrieves the desired (img) display object to be converted. The transformed image object named img is assigned to the image object named Lab. Color spaces in OpenCv are represented by 3-Dimensional bands.

In line 12, cv2.split function in OpenCv splits the image to Bands of the color spaces. It takes the image object (Lab variable) to be separated into bands as a parameter. The bands of Lab image space are a-Dimension, Luminance, and b-Dimension. The bands that are separated here are assigned to variables named a-dim, luminance and b-dim.

In line 14, a band is a 2-dimensional array. To use for the next computations and to solve the array size mismatch problem, a 3-band temporary image object is created using the only a-dim band of the above image.

In line 16 and 17, the image must be a grayscale image for thresholding. Since there is no conversion from LAB color space to grayscale, we first convert the image to BGR, then convert the resulting image to grayscale.

In line 19, we use a function named cv.threshold for thresholding. The first argument is the source image that must be a grayscale image. We already obtained the grayscale image in line 17. The second argument is the threshold value which is used to classify the pixel values. The third argument is the maxVal which represents the value to be given if the pixel value is more than (sometimes less than) the threshold value. OpenCV provides different styles of thresholding and it is decided by the fourth parameter of the function. We use the Otsu algorithm to find the optimal threshold value. Inverse binary thresholding (cv2.THRESH_BINARY_INV) is just the opposite of binary thresholding. The destination pixel is set to zero if the *corresponding source pixel is greater than the threshold value and to max value, if the source pixel is less than the threshold. The formula is as follows:*

$$\texttt{dst}(x,y) = \begin{cases} 0 & \text{if } \texttt{src}(x,y) > \texttt{thresh} \\ \texttt{maxval} & \text{otherwise} \end{cases} \tag{6.1}$$

In lines 22 and 23, we smooth the image in order to reduce noise. The median filter replaces each pixel with the median of its neighboring pixels (located in a square neighborhood around the evaluated pixel). The size of the median filter is 3x3. We also use the functional morphology to apply Morphological Transformation such as opening, closing, morphological gradient, etc. we performed the closing operation. It is useful to remove small holes (dark regions). After the execution of this line, the image object called median was obtained. In line 26, the unwanted pixels obtained after the threshold process which cannot be removed by the median filter have been removed with the dilate morphological function in OpenCv. Dilation is the opposite of erosion. Generally, in cases like noise removal, erosion is followed by dilation. Because, erosion removes white noises, but it also shrinks the object. Therefore, we dilate it. Since noise is gone, they won't come back, but our object area increases. It is also useful in joining broken parts of an object.

In line 29, function connectedComponentsWithStats retrieves useful statistics about each connected component. After dilation in the previous operation, in line 26, we may have several components in the image. We are interested in the biggest object which is ROI (region of interest). Image with 4 or 8-way connectivity - returns N, the total number of labels [0, N-1] where 0 represents the background label. In our segmentation, we used 4 connectivity. If the 4-pixel neighborhood is similar to the other 4 pixels, which is a neighbor to a pixel, then we have considered it to be a component of the object.

In line 32, we define the pixel number of the smallest object as 100000 pixels by experimental tests.

In line 33, num, zeros return a new array of given shape and type, filled with zeros. That is a black image.

In lines 35-37, All the objects are compared with each other and the small ones are excluded. and as a result, a mask that specifies only for the object to be segmented in the image variable named img2 is obtained.

In line 43, we perform LOGICAL AND operation between the original image and img2 which is desired to be segmented that is found in the above loop. The function bitwise_and calculates the per-element bit-wise logical conjunction when two images have the same size.

In line 46, the show displays the image, write saves image data to the file specified by filename.

In line 48, the pyplot class of the matplotlib library which is imported in line 15 is used below begins with plt. The desired image to be shown is given to the subplots as the parameter respectively.

6.4. EXPERIMENTAL TESTS

6.4.1. Field-Based Images

The segmentation process performs well for some field-based images. However, if the background contains nature objects, such as flowers, stones, leaves, etc. then our script cannot make segmentation very well. Another reason is the light and the color of the butterflies. Some examples for successful and unsuccessful segmentations are shown in Figure 6.6 and 6.7, respectively.

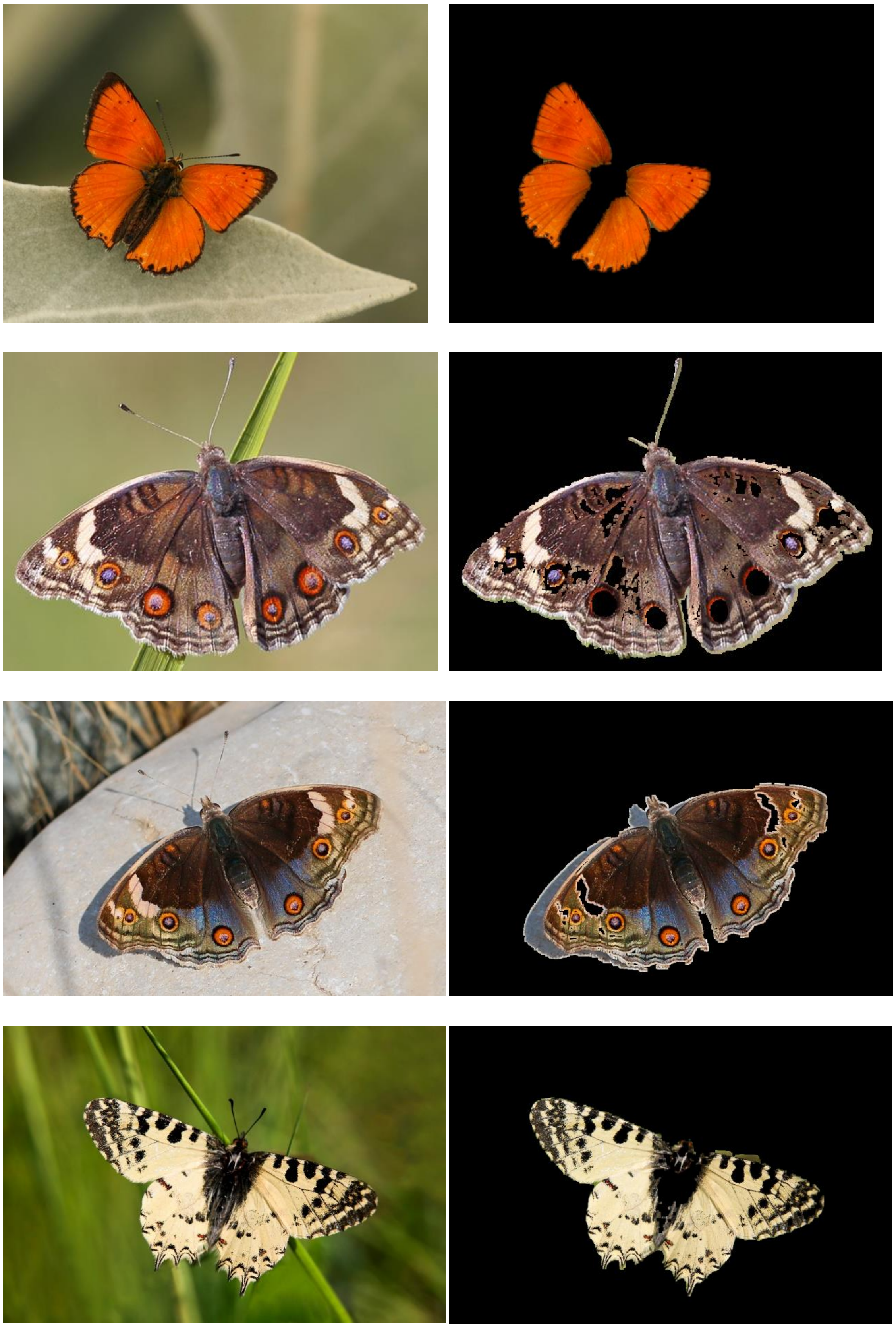


Figure 6.6. Successful segmentations.

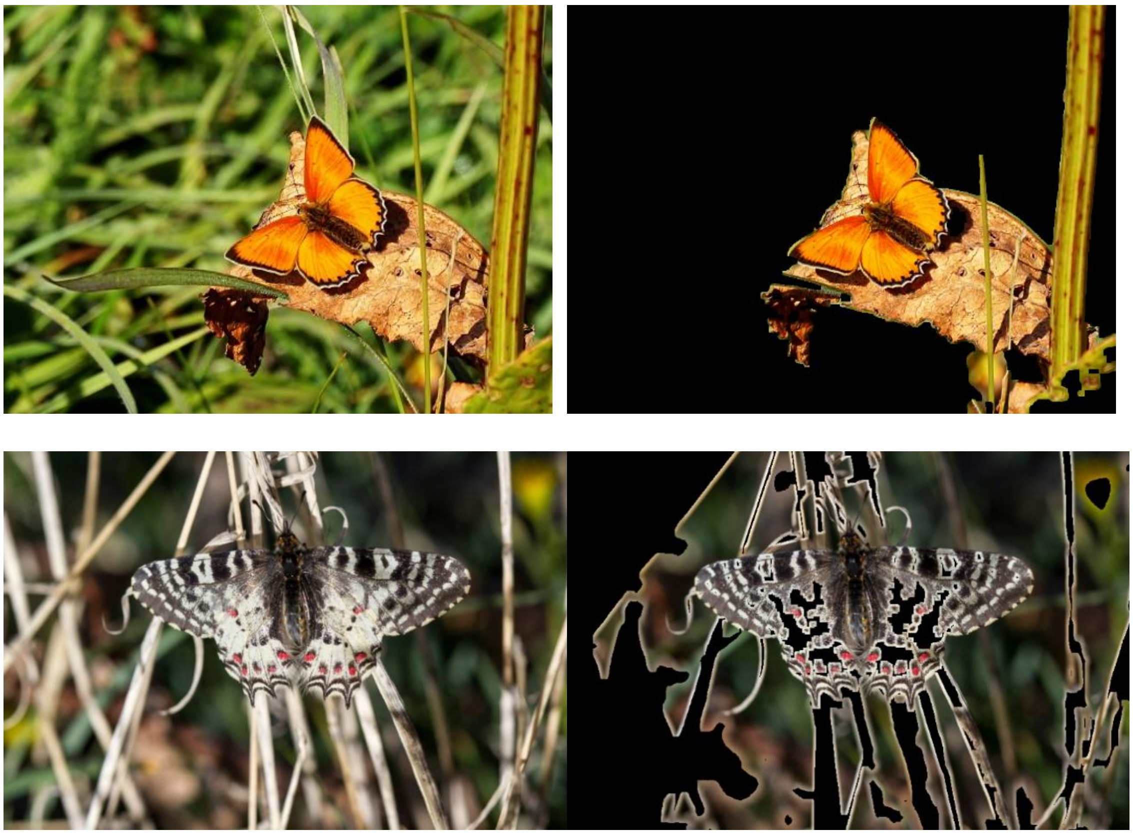


Figure 6.7. Unsuccessful segmentations.

6.4.2. Lab-Based Images

We use different color transformation in lab-based images. The following statements are replaced instead of lines 11,12,14,16,17 of field-based python code, respectively. Here, we use the HSV color space instead of the CIE-Lab color space.

```
Hsv= cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
Hue,Sat,Val = cv2.split(Hsv)
tmpHsv=cv2.merge([Sat,Sat,Sat])
tmpHsv= cv2.cvtColor(tmpHsv, cv2.COLOR_HSV2BGR)
HsvGray=cv2.cvtColor(tmpHsv, cv2.COLOR_BGR2GRAY)
```

The results of all image preprocessing steps are shown in Figure 6.4 are illustrated in Figure 6.8. We show some successful segmentation of lab-based butterfly images in Figure 6.9.

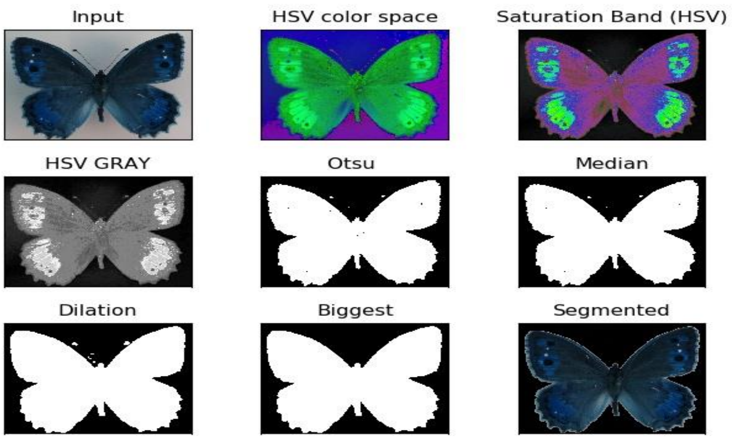


Figure 6.8. All steps of the segmentation.

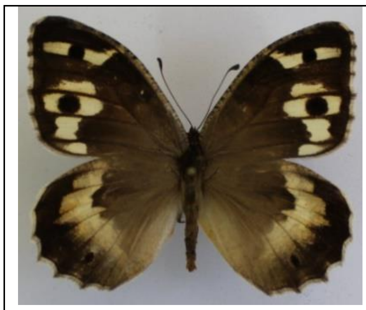
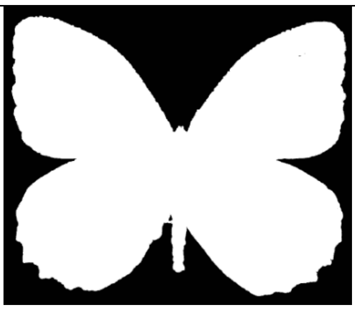
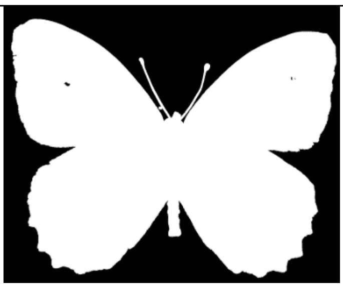
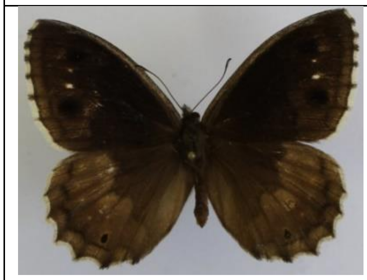
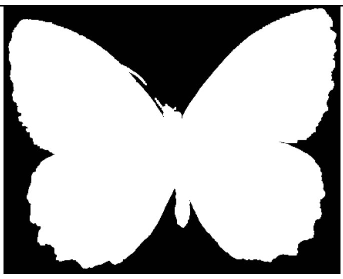
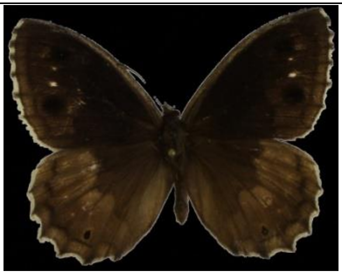
		
Chazara anthe		
		
Chazara briseis		
		
Hipparchia statilinus		

Figure 6.9. Successful segmentation of lab-based images.

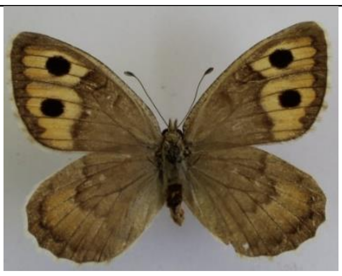
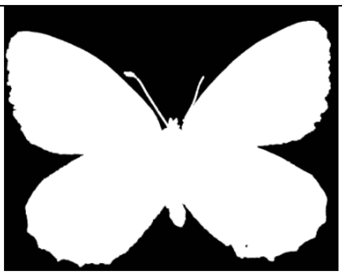
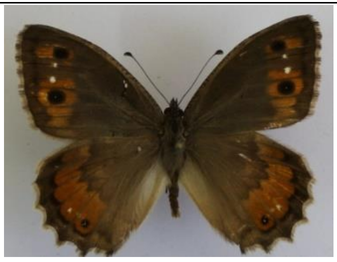
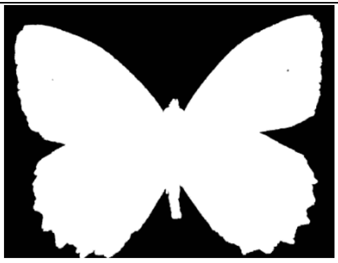
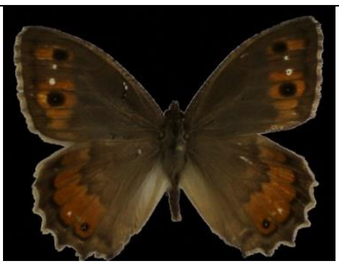
		
Psedochozara beroe		
		
Psedochozara pelopea		

Figure 6.9. (continuing).

PART 7

LOCAL BINARY PATTERN

Local binary patterns (LBP) [30] represent one prominent texture descriptor that has shown effective results in computer vision applications. Their histogram can be used as a powerful local pattern descriptor. These local patterns describe micro-structures (e.g., edges, corners, flat region) and their underlying distribution is estimated by the histogram. The histogram of the image represents local structures extracted from the entire image by combining both structural and statistical information. Figure 7.1 shows an example for evaluating the feature histogram representing the butterfly image where the butterfly image is represented by concatenating a set of local LBP histograms. The success of the LBP in these applications motivated us to utilize it in the butterfly recognition problem. There are a number of the LBP variants proposed recently for specific problems.

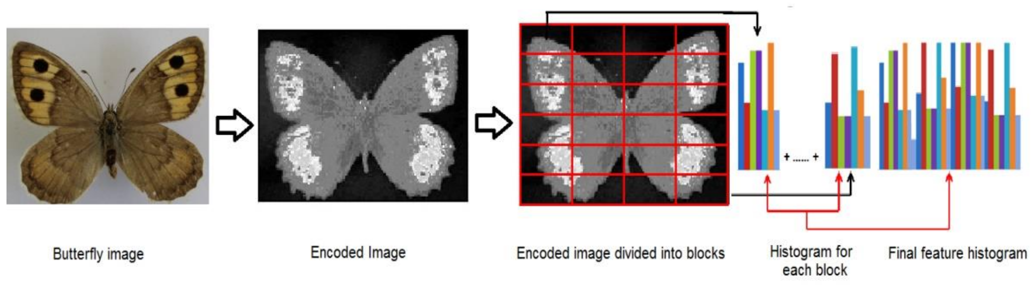


Figure 7.1. The steps for obtaining the final feature histogram representation using LBP.

7.1. THE BASIC LBP

The basic LBP operator was introduced by M. Pietikäinen [31] which characterizes the spatial structures of local texture patterns in images using a 3×3 square patch around a central pixel as illustrated in Figure 7.2. The LBP is defined as a set of binary comparisons of pixels intensities between the center pixel and its eight surrounding pixels. The central pixel with intensity *gc* is compared with each of its eight neighbors, *gi* (*i* = 0, 1, ..., 7) and a binary value is produced according to the following threshold function

$$s(g_i - g_c) = \begin{cases} 1, & if \;\; g_i \geq g_c \\ 0, & if \;\; g_i < g_c. \end{cases} \tag{7.1}$$

Then, the values of the thresholded pixels are weighted by a factor of 2^i and summed to obtain an LBP number representing the texture unit:

$$LBP = \sum_{i=0}^{7} s(g_i - g_c) \times 2^i. \tag{7.2}$$

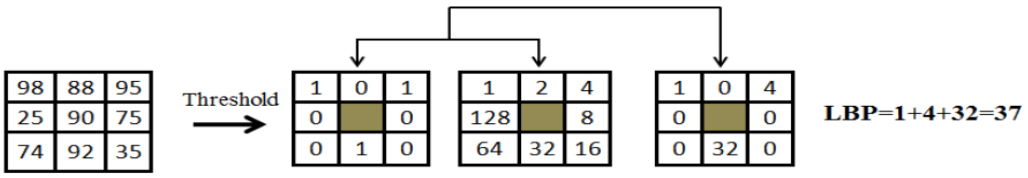


Figure 7.2. Basic LBP computation for a 3 × 3 image patch and the corresponding LBP code.

The basic LBP shows high discrimination ability, success in various texture classification, and works well for extracting local structures from images. However, this operator is sensitive to noise and image rotations.

7.2 UNIFORM LBP

Ojala et al. [32] introduced two useful extensions to the basic LBP descriptor First, by defining the LBP descriptor for any spatial resolution using P equally spaced pixels on a circle of radius R that form a circularly symmetric set of neighbors. Second, by defining a set of fundamental binary patterns, so-called ”uniform” patterns, that occur more frequently than others. The occurrence histogram of these patterns is proven to be a very efficient texture feature and can be used to characterize image patches containing structural information such as edges, spots, and corners. A local pattern is called uniform if it has at most two spatial transitions from 0 to 1 or from 1 to 0. For example, 00011110 and 11110011 (2 transitions) are uniform patterns; while 10010010 and 10100010 (5 transitions) are not. The formulation of the uniform LBP is as follows:

$$\mathrm{ULBP}_{P,R}(x_c, x_y) = \sum_{i=0}^{P-1} s(g_i - g_c) x 2^i, \tag{7.3}$$

$$s(g_i - g_c) = \begin{cases} 1, & \text{if } g_i \geq g_c, \\ 0, & \text{if } g_i < g_c, \end{cases} \tag{7.4}$$

The uniform patterns can reduce the length of histogram vectors. Figure 7.3 shows three circularly symmetric neighbor sets for different values of P and R.

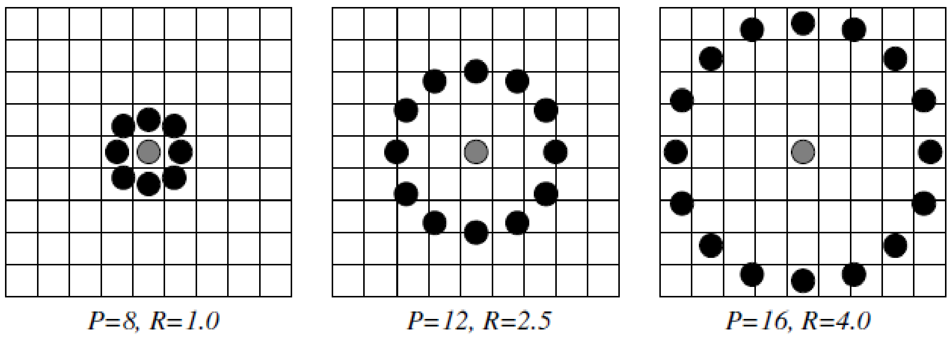


Figure 7.3. Circularly symmetric neighbor sets.

7.3. CENTER SYMMETRIC LBP

Center symmetric LBP (CSLBP) operator M. Heikkila et al. [33] was introduced as a region descriptor that combines the strengths of the well-known the scale-invariant feature transform (SIFT) technique and the LBP. The CSLBP operator replaces the gradient features in the SIFT operator with a local binary pattern for each pixel. Further, within the CSLBP, instead of thresholding each pixel against the central pixel as the basic LBP, only horizontal, vertical, and diagonal pixel comparisons are used as shown in Figure 7.4

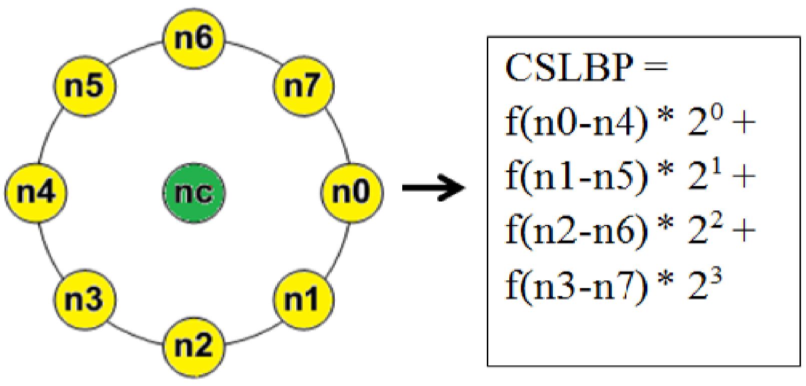


Figure 7.4. Computing the CSLBP pattern for an 8 neighborhood of pixels.

Mathematically, the CSLBP operator is defined for a center pixel with coordinates (x_c, y_c) and a set of P equally spaced pixels on a circle of radius R as follows:

$$CSLBP_{P,R,T}(x_c, x_y) = \sum_{i=1}^{\frac{P}{2}-1} s\left(g_i - g_{i+\left(\frac{P}{2}\right)}\right) x2^i, \tag{7.5}$$

$$s(n) = \begin{cases} 1, & if\ n > T, \\ 0, & otherwise, \end{cases} \tag{7.6}$$

where g_i and $g_{i+(P/2)}$ are the gray values of the center symmetric pairs of pixels in the neighborhood and the threshold value T is set to 1% of the pixel value range. Generally, $2^{P/2}$ distinct patterns can be obtained for a set of P neighboring pixels. For a neighborhood of P = 8 pixels and a radius R = 1 pixel, the CSLBP obtains one of 2^4 = 16, (i = 0, 1, ..., 15) possible distinct patterns. As a result, the dimensionality of the resulting feature histogram is reduced significantly from 2^8 = 256 to 2^4 = 16 dimensions. The resulting features are claimed to be robust on flat image regions, tolerant to illumination variations, and computationally efficient which makes the LBP possible to analyze images in real-time. M. Heikkila et al. [33].

PART 8

FEATURE EXTRACTION

Features can be global or local. Global features based object recognition covers the entire image or a big portion of it, such as eigenspace matching and color histograms. Local features are robust to occlusion, background deformation, and viewpoint change. Local features include edge, corner, entropy, curvature, region, ridge, etc.

In the preprocessing step, images are resized to 256×256 pixels. The binary image is divided into two groups as shown in Figure 8.1, the white pixels are effective pixels after segmentation and the black pixels represent the background. All the feature extraction processes are based on effective pixels. ULBP features computed over the whole butterflies' wings represent only the micropatterns. All the ULBP values can composite a texture spectrum, called texture pattern matrix or ULBP matrix.

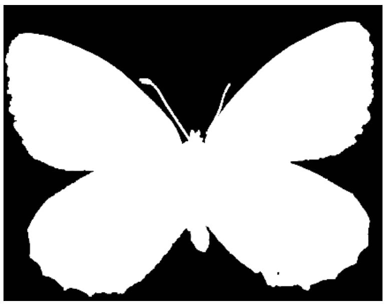


Figure 8.1. After segmentation.

The matrix shows the distribution of gray levels and the main texture information of the butterflies' image. The butterflies' images are divided into 64 sub-blocks for feature extraction. This representation extracts local textures from images. Five texture features are extracted from the ULBP matrix. These are average (F1), deviation (F2), energy (F3), entropy (F4) and correlation (F5). Let f(x,y) be the texture pattern of the ULBP matrix of size M×N. The six features are listed as follows:

$$F1 = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x, y) \tag{8.1}$$

$$F2 = \sqrt{\frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}(f(x, y) - \mathrm{AVE})^2} \tag{8.2}$$

$$F3 = \sqrt{\frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f^2(x, y)} \tag{8.3}$$

$$F4 = -\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\left(\frac{f(x, y)}{\mathrm{ENE}}\ln\left(\frac{f(x, y)}{\mathrm{ENE}}\right)\right) \tag{8.4}$$

$$F5 = -\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\frac{xyf(x, y) - \mu_x\mu_y}{\sigma_x\sigma_y} \tag{8.5}$$

where K represents the maximum value for the ULBP matrix, μ is the mean and σ is the standard deviation. We also use six color features of butterfly images.

<u>HSV (Hue-Saturation-Value) color feature</u>: The color feature is insensitive to changes in size and direction of regions. However, it suffers from the influence of illumination variations. For the color feature extraction, the original RGB (Red-Green-Blue) color space is first transformed into HSV (Hue-Saturation-Value) space and only the hue and saturation components are used to reduce the impact from lighting conditions. We then obtain mean hue, the standard deviation of hue, mean saturation, the standard deviation of saturation, mean value, and standard deviation of value in HSV color space. In Figure 8.2, it is shown that a butterfly image, its ULBP, HSV, and HSV components.

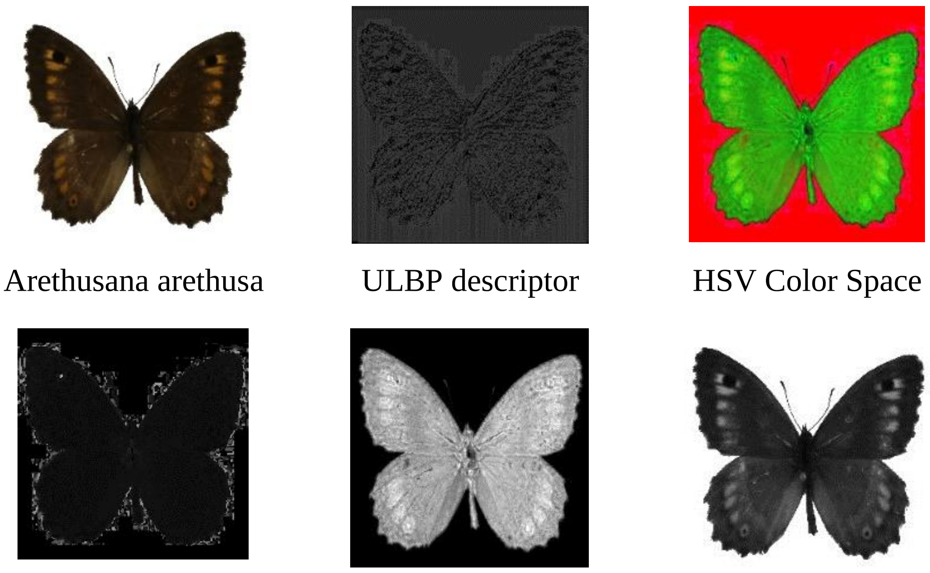


Figure 8.2. ULBP and HSV components of a given butterfly.

We have a total of 11 features of butterflies' images. The list of the values of all features are listed in Table 8.1.

Table 8.1. Features of 140 butterflies.

samples	P=8 and R=3					HSV Color Space						time (sec)
	F1	F2	F3	F4	F5	hue mean	hue std	sat mean	sat std	val mean	val std	
1	44.24	18.39	2.37	4.12	88.28	15.15	23.87	89.13	86.54	143.92	104.34	32.37
2	43.16	18.83	2.37	4.30	71.57	16.89	25.39	97.66	85.68	133.15	103.11	37.84
3	42.76	18.86	2.37	4.39	66.09	17.64	27.36	98.32	85.06	131.25	102.49	39.56
4	43.33	18.84	2.37	4.27	73.90	17.28	27.07	94.93	85.10	134.53	103.33	37.57
5	43.26	18.68	2.37	4.30	73.48	17.74	27.48	95.05	85.45	135.39	103.30	38.30
6	41.32	17.46	2.37	4.66	50.19	16.08	30.32	99.34	84.43	131.62	100.64	27.30
7	41.28	17.48	2.37	4.67	49.61	16.35	30.49	99.41	84.09	130.94	100.83	24.73
8	41.72	17.72	2.37	4.62	55.28	16.70	27.92	96.33	84.25	132.56	102.85	24.23
9	43.09	18.32	2.37	4.38	72.66	17.19	28.87	93.36	84.39	134.57	104.96	30.29
10	42.92	18.46	2.37	4.39	69.74	17.73	29.66	94.24	84.21	133.22	104.76	28.93
11	40.52	18.53	2.39	4.82	39.83	20.78	29.21	95.56	74.77	135.93	94.84	27.70
12	40.62	18.59	2.37	4.80	37.50	20.92	29.30	95.30	74.86	135.91	95.07	28.27
13	40.46	18.63	2.37	4.81	35.30	20.46	28.92	97.12	74.68	134.82	94.25	29.29
14	41.01	18.59	2.40	4.74	45.60	19.52	28.43	94.56	75.40	137.64	95.11	29.77
15	41.30	18.77	2.37	4.67	46.41	20.77	30.12	94.94	76.42	136.29	96.28	33.96
16	41.16	18.85	2.37	4.70	55.06	20.71	29.74	95.74	75.89	134.58	96.05	49.74
17	41.91	19.30	2.37	4.49	53.21	19.56	28.57	109.01	89.47	140.56	93.98	45.06
18	42.10	19.40	2.37	4.44	55.57	19.97	28.92	108.95	90.24	139.22	95.29	46.60
19	42.25	19.41	2.41	4.41	56.59	19.81	28.43	109.03	89.69	138.86	94.86	39.84
20	42.55	19.25	2.37	4.38	61.97	19.18	28.00	105.73	90.57	142.02	95.82	38.04
21	40.82	17.35	2.37	4.77	43.23	18.11	27.41	99.51	90.54	155.82	91.31	23.35
22	41.10	18.08	2.43	4.74	43.69	18.18	27.91	107.22	92.65	147.12	94.77	28.76
23	40.54	17.45	2.39	4.82	46.46	17.21	26.38	104.26	90.93	150.57	91.81	23.89
24	40.41	17.63	2.39	4.84	42.41	18.65	28.16	105.44	90.45	151.55	90.51	28.57
25	40.30	17.72	2.37	4.86	38.79	18.76	27.85	105.69	90.69	151.78	90.53	25.73
26	41.05	17.62	2.38	4.74	51.69	17.27	26.81	100.92	91.09	156.66	90.80	24.92
27	40.35	17.82	2.39	4.85	43.30	18.14	27.14	109.18	91.20	141.51	95.18	27.18

Table 8.1. (continuing).

28	40.86	17.92	2.40	4.77	41.46	18.29	28.84	105.33	91.98	148.26	94.45	27.99
29	40.97	18.05	2.40	4.75	42.22	18.12	28.27	106.80	91.85	146.61	94.53	28.28
30	41.33	18.09	2.37	4.70	48.56	18.14	28.05	105.72	92.80	148.63	94.94	31.93
31	42.91	18.47	2.37	4.39	69.48	19.70	29.20	79.73	74.93	153.64	98.79	38.71
32	42.34	18.62	2.37	4.49	61.12	20.38	29.31	83.26	74.59	149.92	98.31	36.98
33	42.26	18.67	2.37	4.51	59.83	20.79	29.26	84.48	74.36	147.84	98.34	41.87
34	41.80	18.88	2.35	4.56	62.78	20.05	27.79	87.66	72.98	142.75	97.82	38.43
35	42.20	18.69	2.37	4.51	70.55	20.24	28.56	85.01	75.22	148.43	98.38	36.79
36	39.72	17.88	2.40	4.86	24.29	19.41	26.62	88.91	71.51	140.25	97.34	25.56
37	39.81	18.01	2.42	4.85	27.17	19.76	27.18	89.11	71.56	138.28	98.21	26.10
38	40.45	17.97	2.37	4.79	44.64	19.43	27.43	86.02	72.68	143.01	98.92	25.90
39	40.50	18.06	2.39	4.78	34.99	19.63	27.53	87.28	72.75	141.65	98.75	26.77
40	41.37	17.97	2.37	4.65	49.51	18.95	28.06	81.57	73.92	149.62	99.43	27.16
41	43.26	18.47	2.37	4.34	74.42	16.57	26.82	89.80	78.53	130.81	105.63	34.48
42	43.59	18.61	2.37	4.25	78.38	16.54	26.99	90.01	79.06	131.73	105.19	36.42
43	43.15	18.85	2.38	4.32	76.72	16.99	26.22	94.08	77.49	123.00	105.13	38.14
44	42.84	18.79	2.39	4.40	80.03	17.26	25.93	97.50	78.98	123.21	103.72	37.27
45	43.22	18.78	2.41	4.32	70.35	16.66	25.95	92.49	77.21	123.84	105.47	38.98
46	42.71	18.85	2.37	4.40	65.44	17.56	25.26	92.55	77.39	122.23	106.51	37.87
47	40.82	18.43	2.37	4.74	40.64	17.70	25.33	95.18	74.10	111.97	105.69	30.74
48	41.15	18.45	2.37	4.71	45.11	17.58	26.17	95.02	76.88	119.98	104.60	29.50
49	41.23	18.55	2.41	4.69	42.69	17.95	26.62	96.10	76.91	118.77	104.45	30.54
50	41.10	18.82	2.37	4.68	56.43	18.21	27.00	99.19	76.63	115.41	103.09	34.44
51	44.61	18.37	2.40	4.09	92.96	24.17	33.98	58.01	70.75	179.13	91.21	44.04
52	45.13	18.15	2.37	3.96	102.00	22.85	33.24	54.96	69.40	182.58	90.26	43.33
53	44.52	18.46	2.39	4.08	93.13	24.54	34.26	57.89	69.40	178.39	90.98	41.25
54	44.77	18.23	2.37	4.06	96.50	23.66	33.53	55.44	68.84	181.35	90.47	42.68
55	44.76	18.13	2.37	4.08	96.92	23.74	34.23	50.89	62.31	183.10	87.68	38.81
56	45.48	17.95	2.37	3.93	108.12	23.46	34.88	47.39	62.43	186.68	88.21	43.69
57	45.07	18.18	2.37	3.99	101.10	24.09	34.67	51.17	63.97	183.45	88.17	43.56
58	44.99	18.21	2.37	4.01	99.68	24.25	34.39	51.98	64.62	183.27	88.21	41.06
59	45.01	18.25	2.40	3.99	98.98	24.42	34.77	50.51	64.20	183.75	89.04	43.52
60	45.40	18.01	2.37	3.94	106.64	24.17	35.38	49.20	64.33	186.32	88.25	43.57
61	41.84	18.91	2.37	4.50	53.48	22.36	28.73	57.86	55.49	161.51	91.66	33.60
62	40.13	18.19	2.37	4.78	31.42	22.01	26.15	52.79	49.00	164.88	88.05	25.50
63	40.05	18.01	2.36	4.80	38.08	22.43	26.89	52.36	49.20	165.04	88.51	27.28
64	42.47	19.33	2.39	4.33	57.82	22.42	27.19	62.25	62.85	159.17	93.58	38.65
65	42.36	19.37	2.37	4.33	61.60	22.78	27.74	62.41	62.47	157.69	93.77	37.62
66	43.17	18.43	2.37	4.30	73.25	21.38	29.48	47.04	53.32	174.51	89.41	30.48
67	43.87	18.71	2.37	4.12	81.78	21.91	29.93	50.70	60.70	173.66	91.02	37.03
68	43.87	18.71	2.37	4.12	81.78	21.91	29.93	50.70	60.70	173.66	91.02	34.97
69	43.08	18.94	2.37	4.27	70.07	22.16	29.28	52.96	58.66	168.86	90.49	37.11
70	43.08	18.94	2.37	4.27	70.07	22.16	29.28	52.96	58.66	168.86	90.49	37.16
71	42.89	18.77	2.37	4.36	68.09	19.52	27.95	71.57	69.89	151.09	100.52	34.53
72	42.06	18.50	2.37	4.52	57.65	18.94	27.25	69.31	66.71	153.44	98.59	30.48
73	42.78	18.73	2.37	4.39	66.80	19.92	28.51	69.42	68.58	154.00	99.35	34.08
74	42.89	18.77	2.37	4.36	68.09	19.52	27.95	71.57	69.89	151.09	100.52	36.17
75	43.06	18.84	2.38	4.33	72.70	19.26	27.45	71.67	70.78	151.38	100.65	36.99
76	43.12	18.92	2.37	4.30	70.77	19.82	28.17	72.15	70.61	149.45	100.95	36.10
77	40.73	18.69	2.37	4.70	38.88	20.30	27.98	73.70	65.22	145.74	98.52	26.09
78	42.20	19.19	2.40	4.42	56.13	20.07	26.77	78.68	73.51	146.67	100.37	35.01
79	41.93	18.93	2.37	4.50	54.59	20.22	28.09	76.92	70.83	147.06	99.62	31.85
80	42.16	19.38	2.37	4.40	56.35	20.59	28.40	77.84	71.15	143.23	100.58	34.60
81	41.38	19.25	2.41	4.56	52.00	20.13	27.41	92.58	75.55	156.60	80.39	30.46
82	42.46	19.55	2.37	4.31	59.74	19.15	26.31	91.81	76.56	157.91	80.67	35.66
83	43.32	19.19	2.37	4.18	72.43	19.55	27.95	87.53	78.40	162.97	82.00	35.21
84	42.54	19.66	2.32	4.27	83.34	20.72	28.41	94.96	76.91	154.91	80.64	36.59
85	42.78	19.55	2.38	4.24	66.68	19.87	27.42	92.11	77.71	155.93	82.88	36.71
86	39.21	18.44	2.36	4.88	34.60	19.27	26.32	96.53	77.81	151.14	82.81	21.99
87	41.23	18.42	2.37	4.61	46.41	17.51	25.78	87.09	79.19	161.39	84.34	23.49
88	40.57	18.71	2.37	4.71	41.11	18.74	26.24	93.91	79.12	154.77	83.67	24.65
89	40.33	18.91	2.37	4.74	33.03	19.37	26.57	96.79	78.73	151.98	83.04	24.84
90	41.14	18.85	2.37	4.62	44.10	18.23	25.68	93.28	79.26	155.81	83.50	25.79
91	42.61	18.99	2.37	4.40	63.52	18.54	25.95	93.08	79.58	140.22	96.17	33.71

Table 8.1. (continuing).

92	42.71	18.93	2.37	4.38	65.08	17.61	24.98	91.76	80.32	142.77	96.31	33.05
93	42.81	19.14	2.41	4.33	61.88	18.54	26.40	91.97	78.20	139.72	95.66	35.45
94	42.86	18.99	2.37	4.35	66.92	17.85	25.46	91.13	78.93	141.57	95.96	33.77
95	42.73	19.06	2.38	4.37	69.87	18.69	26.55	92.50	79.57	141.48	95.53	35.05
96	41.99	18.70	2.37	4.55	56.03	17.74	24.86	93.20	79.31	140.73	95.63	30.35
97	42.86	18.99	2.37	4.35	66.92	17.85	25.46	91.13	78.93	141.57	95.96	34.77
98	41.66	19.06	2.37	4.57	50.55	18.78	26.85	100.47	79.41	132.61	94.70	31.56
99	42.29	19.21	2.37	4.43	58.51	18.72	27.08	99.96	78.99	132.29	94.65	35.02
100	42.96	19.28	2.37	4.28	67.28	18.57	27.30	97.28	79.89	136.03	95.37	36.68
101	39.85	18.57	2.31	4.88	58.71	20.74	26.81	103.06	76.81	138.81	85.17	24.87
102	40.34	18.30	2.38	4.83	34.78	19.97	27.07	97.79	77.79	143.80	86.92	24.77
103	40.85	18.38	2.40	4.75	44.37	19.45	27.10	94.52	77.74	148.35	86.31	26.39
104	40.06	18.54	2.37	4.85	46.22	19.91	26.33	101.61	76.85	139.11	86.11	24.59
105	39.95	18.61	2.37	4.86	35.20	19.11	24.78	102.37	76.48	138.24	85.56	25.15
106	42.05	18.75	2.40	4.54	53.84	19.52	27.48	103.31	85.38	139.31	92.30	30.54
107	43.04	18.72	2.37	4.36	70.43	18.80	27.70	98.29	86.62	141.80	95.61	34.27
108	42.71	18.83	2.34	4.41	90.15	18.56	26.79	99.76	85.78	140.16	94.74	33.57
109	42.79	18.93	2.36	4.39	75.20	19.64	28.33	102.43	86.85	138.73	94.67	35.13
110	43.09	18.98	2.40	4.31	76.12	19.38	28.11	99.07	85.95	141.26	94.52	36.32
111	43.37	18.66	2.37	4.27	75.12	14.80	24.04	94.56	81.47	121.12	110.19	33.34
112	43.13	18.93	2.37	4.29	70.74	14.52	22.88	94.68	79.94	119.50	108.71	36.23
113	42.91	18.99	2.37	4.32	67.56	15.74	23.65	98.07	80.30	116.79	108.15	34.76
114	42.35	19.39	2.37	4.37	58.79	16.49	24.39	102.96	78.26	107.97	105.67	36.82
115	42.46	19.23	2.37	4.37	60.78	16.09	24.02	102.37	79.28	110.83	106.16	35.84
116	43.87	18.79	2.37	4.15	81.47	14.64	23.87	93.89	79.57	119.01	110.71	36.04
117	43.81	18.82	2.37	4.15	80.52	14.33	24.63	92.41	78.27	117.40	111.43	36.10
118	43.83	18.83	2.37	4.14	80.69	14.78	23.28	92.12	78.68	118.19	111.89	35.93
119	44.03	18.68	2.37	4.13	84.12	14.46	23.03	90.99	78.76	120.04	112.05	35.69
120	43.24	19.28	2.37	4.20	70.93	14.96	23.43	100.71	78.15	108.75	108.20	37.12
121	43.48	19.17	2.36	4.20	85.56	15.84	26.44	104.00	86.18	118.20	107.22	37.02
122	41.92	17.50	2.37	4.61	58.82	14.23	23.70	105.06	88.29	120.59	108.22	23.85
123	41.89	17.65	2.44	4.62	46.58	14.80	25.05	106.54	88.25	119.62	107.56	23.87
124	43.69	18.80	2.40	4.19	81.16	16.63	30.05	103.08	88.29	123.83	107.96	34.66
125	44.21	18.73	2.37	4.09	86.27	14.69	25.12	98.04	87.12	123.63	111.20	36.43
126	44.04	18.76	2.37	4.13	83.84	15.05	25.49	102.50	88.83	122.75	109.79	36.55
127	43.99	18.98	2.37	4.11	82.26	16.05	26.85	101.04	87.82	123.09	108.69	36.71
128	43.87	19.04	2.37	4.13	80.30	16.67	28.61	100.88	87.01	123.68	107.37	38.09
129	43.88	19.15	2.37	4.10	79.95	16.58	28.09	98.75	85.40	123.76	107.09	38.82
130	43.73	18.94	2.37	4.19	78.84	15.17	25.67	97.93	85.60	125.52	107.66	35.84
131	43.67	18.92	2.42	4.17	78.63	15.12	22.61	101.87	83.15	109.73	112.25	38.05
132	43.47	19.11	2.38	4.19	82.91	15.29	24.57	102.21	81.07	108.66	109.50	37.15
133	43.60	19.07	2.37	4.16	76.64	15.44	23.88	101.84	82.37	112.42	109.39	37.46
134	43.65	19.05	2.37	4.16	77.30	15.82	24.50	99.91	81.91	113.13	110.03	37.71
135	43.40	19.20	2.42	4.18	79.19	15.50	23.43	102.61	81.67	109.52	109.19	41.78
136	43.73	18.87	2.37	4.16	79.21	15.62	24.21	99.36	83.86	117.17	110.71	39.74
137	43.34	19.26	2.37	4.18	72.43	15.78	23.82	103.43	81.72	109.34	108.71	42.89
138	43.94	18.83	2.41	4.13	88.38	15.56	26.50	98.28	83.27	116.66	111.25	39.19
139	43.79	18.84	2.37	4.16	80.07	15.25	24.96	98.58	82.31	115.05	110.94	37.95
140	43.65	18.98	2.37	4.17	77.59	15.41	26.21	101.40	82.66	112.40	110.15	38.05

We normalized both input in Table 8.1 and output data sets to a range of the interval [-1, 1] before the training process using the following equation.

$$x_{normal} = 2\frac{x-x_{min}}{x_{max}-x_{min}} - 1 \qquad (8.6)$$

Levenberg–Marquardt (LM) learning algorithm was used for the training of the networks. The input and target data sets were divided into three parts randomly for training, validation, and testing. We used the training set which consists of 70% (98) of the data set to compute the gradient and update the weights and biases of the network. We used 15% (21) of the data set as validation to estimate the weights and biases to measure network generalization and to decide when to terminate training. We used the other 15% (21) of the data set for validating the weights and biases to show the effectiveness of the termination criterion and determine the performance of the network on new data sets that are not trained during learning. A minimum gradient of 10^{-10} and a maximum number of epochs of 1000 were used in the training process. No transfer function was used in the neurons in the input layer, while neurons with tangent sigmoid (tansig) and linear (purelin) transfer functions were used in the hidden layer(s) and output layer, respectively. We used the different networks with single and double hidden layer topologies and the number of neurons was varied from 5 to 15. The network has two hidden layers with 10 neurons trained using the LM algorithm gave the best performance.

We developed the ANN model in MATLAB version R2014b using the Neural Network Toolbox. There exist three layers (input, hidden and output layers), eleven input parameters and one output parameter in the network as illustrated in Figure 8.3.

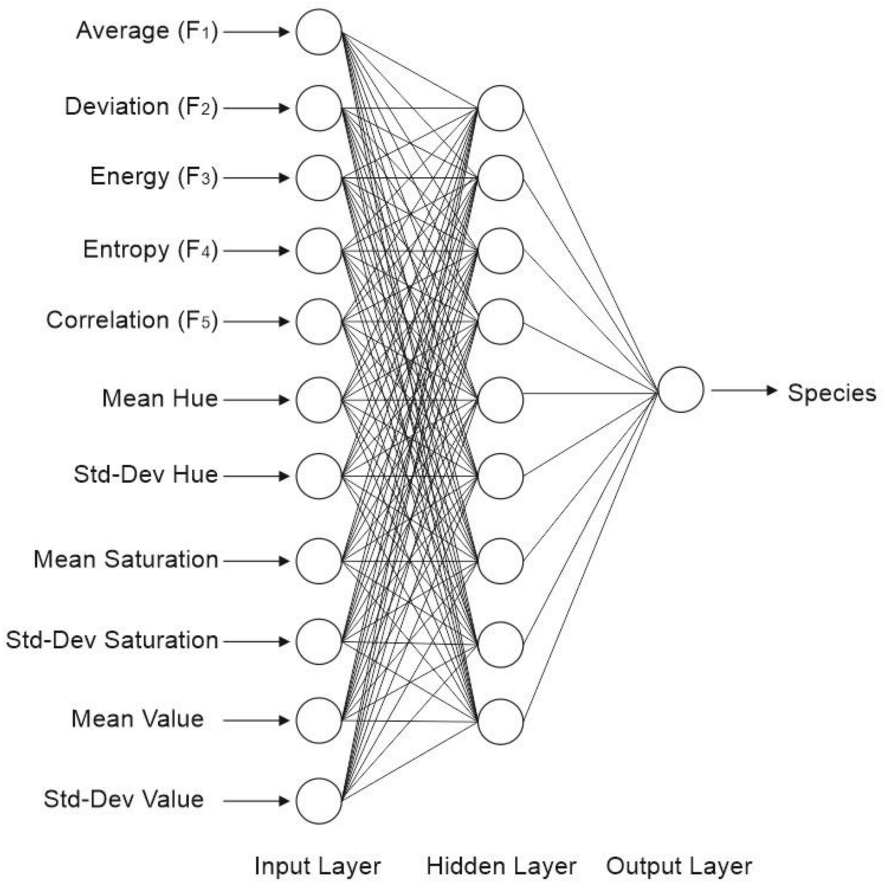


Figure 8.3. Schematic diagram of an ANN for prediction of butterfly species with one input layer, one hidden layer, and one output layer.

The view of the ANN model designed in Matlab is shown in Figure 8.4.

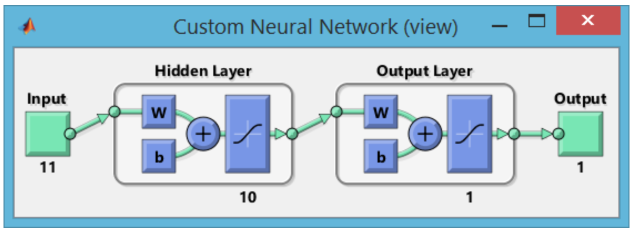


Figure 8.4. The view of the ANN in Matlab.

R measures the correlation between the predicted and the actual values. R-value of 1 and 0 means a close and random relationship, respectively. The correlation coefficient (R) between the predicted and the actual butterfly species is shown in Figure 8.5.

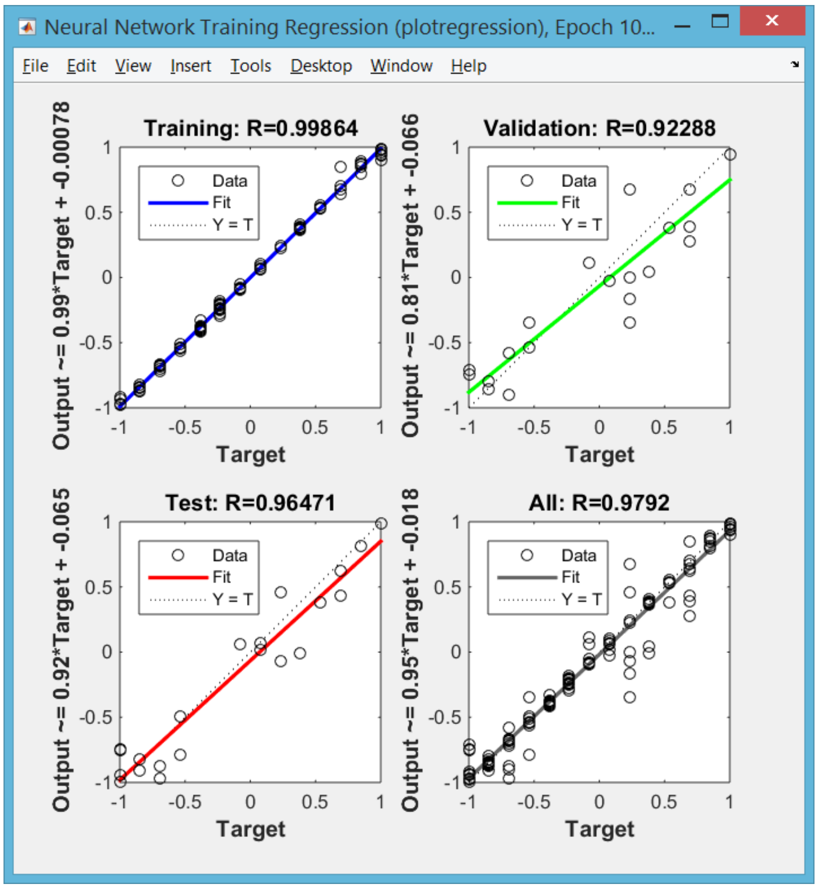


Figure 8.5. The performance values of the ANN model with one hidden layer and 10 neurons.

PART 9

DEEP NEURAL NETWORKS

Deep learning, especially in recent years, has become key research for applications based on artificial intelligence [32]. Due to the serious achievements in the field of computer vision [39], natural language processing [40], and speech recognition [41], the rate of use increases day by day. Deep learning algorithms, which are based on Artificial Neural Networks that are based on neuron cells in the human brain, come to the forefront with its success in the learning phase. Deep learning algorithms can solve the problem of feature extraction and selection by automatically removing the distinguishing features from the given input data. This remark has attracted the attention of many researchers in the field of computer science.

Convolutional Neural Networks (CNNs) are deep learning models, which are mainly used to image classification, similarity detection and object recognition [42-43]. CNN has the ability to identify Faces [44], Persons [45], and Signs [46], and so on. As shown in Figure 9.1, CNN consist of 5 layers which are Input, Convolution, Pooling, Fully Connected and Output.

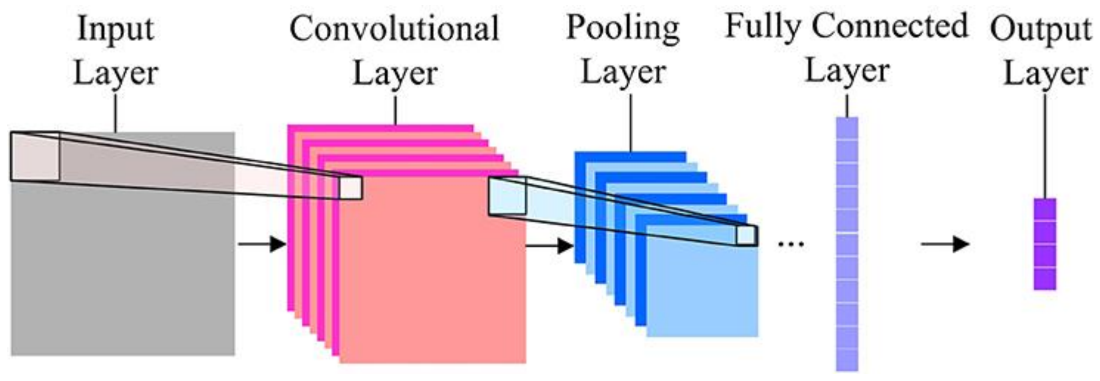


Figure 9.1. CNN architecture.

In the Input layer, images are taken to model as input data. If it is necessary, some preprocessing algorithms such as scaling and noise reduction are applied to the input data. In the Convolution layer, the Convolution process applied to the input data through filters as feature selection. The filter size setting up as 1x1, 3x3, 5x5 and sometimes 7x7. Figure 9.2 shows example sobel filter applying process to sample image data. This process continues to overall images for all filters. Finally, features are obtained end of this process. If the input images have 3 channels (RGB), convolution process applied to each channel.

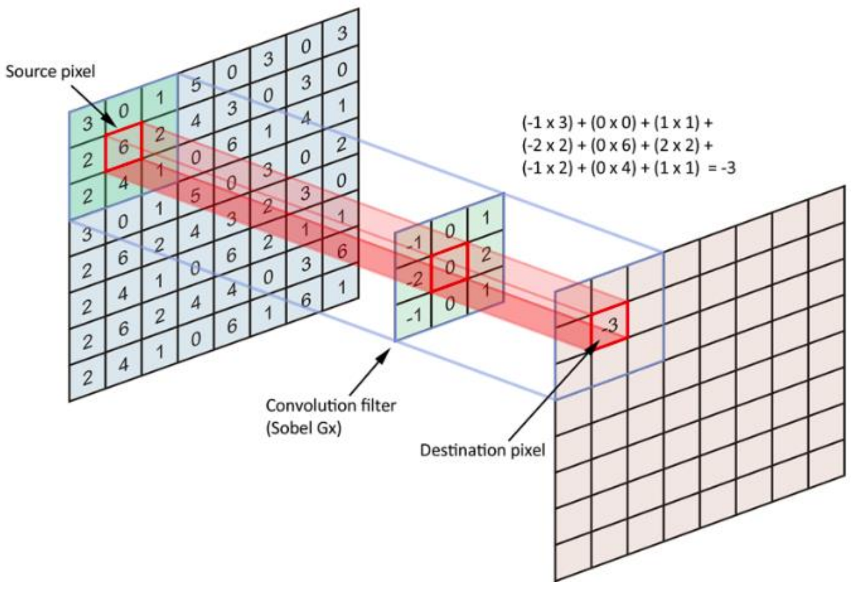

Source pixel
(-1 x 3) + (0 x 0) + (1 x 1) +
(-2 x 2) + (0 x 6) + (2 x 2) +
(-1 x 2) + (0 x 4) + (1 x 1) = -3
Convolution filter
(Sobel Gx)
Destination pixel

Figure 9.2. Convolution process.

The padding process determines the process for the pixel information to be inserted in the corners of the input matrix. Stride shows how many steps the window will be shifted in each step. After convolution processing to nxn size input matrix resulting from the applied fxf sized filter on the image is calculated by the following equation.

$$n_x, n_y = \left[\frac{n+2p-f}{s} + 1\right] \tag{9.1}$$

Where p padding size, s stride number. The pooling layer is used to reduce the size of the input matrix. By keeping the channel number constant, the width and height size can be reduced. In this layer, maximum and average pooling is the common two approaches. Forgiven the pool size (2,2) for example in Figure 9.3 maximum number or average of matrices is determined as representing the value of the pool.

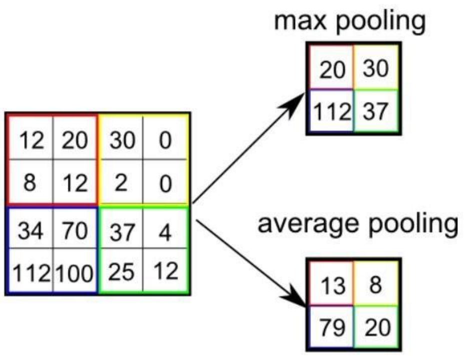


Figure 9.3. Max and average pooling.

In the Full-Connected layer, the data from the pooling layer is transformed into a single dimension. Because each neuron is interconnected, it is called fully connected. In this layer, the classification process is carried out and activation functions such as RelU and sigmoid are used. In the output layer, using the Softmax function, it produces the probability-based loss value using the score values generated by the model. Figure 9.4 shows the output layer for a two-class problem.

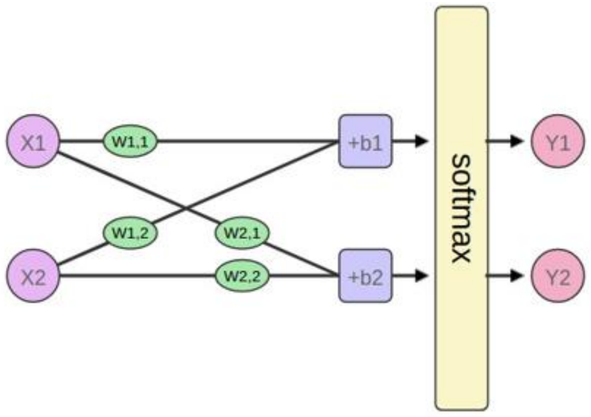


Figure 9.4. Softmax function.

The dropout layer that is used in cases where the model performs over-fitting (memorizing) makes the process of eliminating some connections in the model that make over-learning. Thus, the network is prevented from memorizing.

The classic CNN architecture consists of two basic parts. The first is the convolution and pooling layers. Features are extracted on the input images in this section. In this section, feature extraction stages are applied in the general area but not the problem. In the second part, the FC layer and the output layer perform problem-specific classification.

9.1. TRANSFER LEARNING

Transfer learning is the rapid implementation of the learning process that takes place in a field in a similar or different field [51]. In this way, the learning process realized on a large amount of data can be applied to different sets of images in smaller sizes. Thanks to its improvement in memory usage and performance, it is especially popular in the field of computer vision. The model, which is trained with a huge amount of training data, is completed by updating the weights in the classification section instead of training from the beginning. There are many studies that have been suggested with this approach, which is commonly called pre-trained models. The most popular models VGG [52], Inception [53], MobileNet [54], ResNet [55] are applied to ImageNet [56] which has more than 1.2 million images and have achieved high accuracy [57]. In this study, various deep learning models are used to classify the images in the data set. Convolution Neural Networks based learning transfer methods were used in the study (VGG16, VGG19, ResNet50). The most widely used models in the literature have been trained on the ImageNet dataset and have achieved high success. In this study, fine-tuned transfer learning methods used for the classification of butterfly images. In this section, 900 sample images are used for each class given in Table 6.1. Split process for training and testing was carried out on 1800-7200 images of 20% -80%. Accuracy is used for classification success for each model. The study was performed using the Python programming language and the Keras library. Deep learning studies have been run on the GPU because they require high memory. The GPU operated has 2560 cores.

9.2. VGG16 - VGG19

The VGG network architecture was introduced by Simonyan and Zisserman in their paper, Very Deep Convolutional Networks for Large Scale Image Recognition [51]. This network is characterized by its simplicity, using only 3×3 convolutional layers stacked on top of each other in increasing depth. Reducing volume size is handled by max pooling. As shown in Figure 9.5 Two fully-connected layers, each with 4,096 nodes are then followed by a softmax classifier. The "16" and "19" stand for the number of weight layers in the network (columns D and E in Figure 9.6):

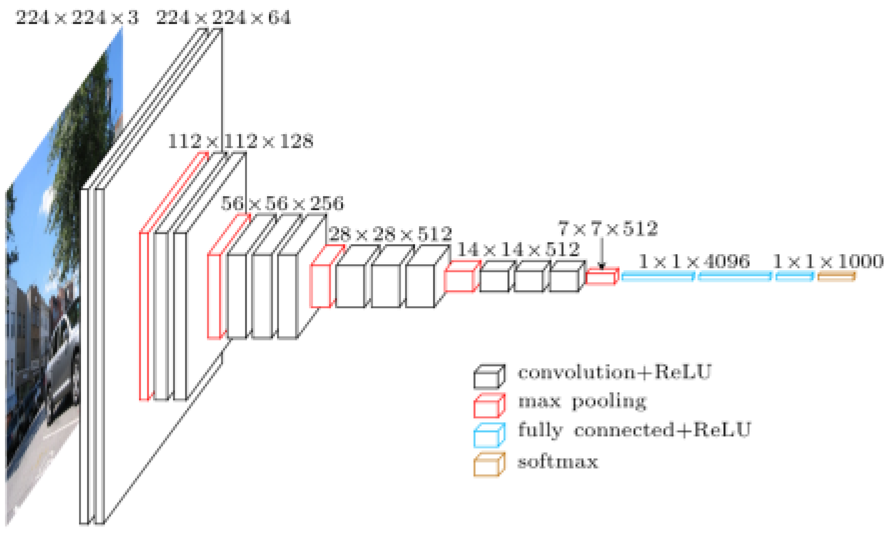
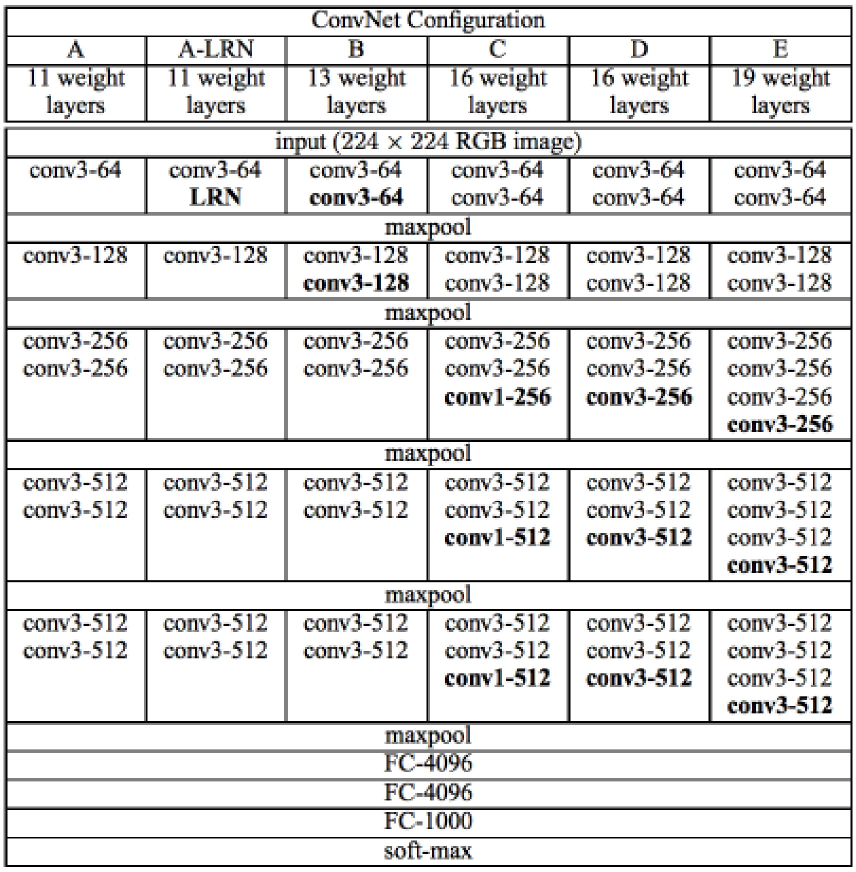


Figure 9.5. A visualization of the VGG architecture.

In 2014, 16 and 19 layer networks were considered very deep (although we now have the ResNet architecture which can be successfully trained at depths of 50-200 for ImageNet and over 1,000 for CIFAR-10).



ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Figure 9.6. ConvNet configurations for VGGNET.

9.3. RESNET 50

ResNet has a different structure than the traditional consecutive network architecture such as VggNet, AlexNet, because has micro-architectural module structure differs from other architectures. It may be preferable to switch to the lower layer by ignoring the change between some layers. This situation was allowed in the architecture of Resnet and the success rate of the network was increased by eliminating the problem of memorizing the network. Resnet50 architecture has a network of 177 layers. In addition to this layer structure, there is information about how inter-layer connections. This model has trained for images that are sized as 224x224x3. Figure 9.7 shows the connection example used in Resnet architecture.

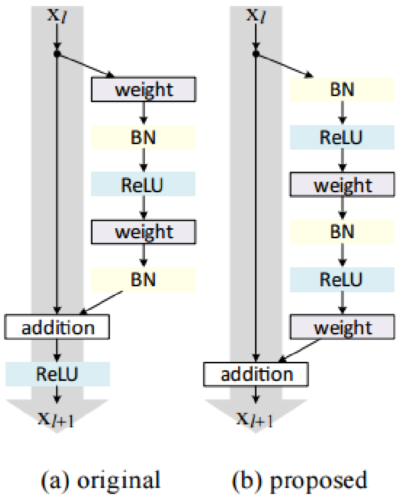


Figure 9.7. Resnet architecture.

9.4. CLASSIFYING OF BUTTERFLY GENUS

In this subsection, to classify the butterfly species, real-world images labeled by users are used. The 42000 images downloaded from the Website were organized to have 10 classes and 17769 images with the highest number of samples. A transfer learning approach has been used since the number of high samples needed by the deep learning algorithms cannot be reached. Since the dimensions of the samples are different and the models used have an input size of 224x224x3, the dimensions have been changed by applying the preprocessing step.

VGG16, VGG19 and ResNet50 deep learning models were used to classify butterfly images. It has been run on the GPU because it takes a long time for all learning and testing to be performed on the CPU. The parameters of the models are as follows.

Table 9.1. The real-world butterfly species images labeled by users.

Parameter Name	**Value**
Input sizes	224x224x3
Max Epoch	100
Batch Size	8
Number of Classes	10
Number of training images	7200
Number of test images	1800
Optimizer	AdaDelta

The accuracy and loss curves obtained after executing the models are given in Figures 9.8-9.10. VGG16 and VGG19 provide approximately the same results for our data set. The ResNet50 model achieves higher success than VGGNET in the training phase, while it has a lower success in the test phase.

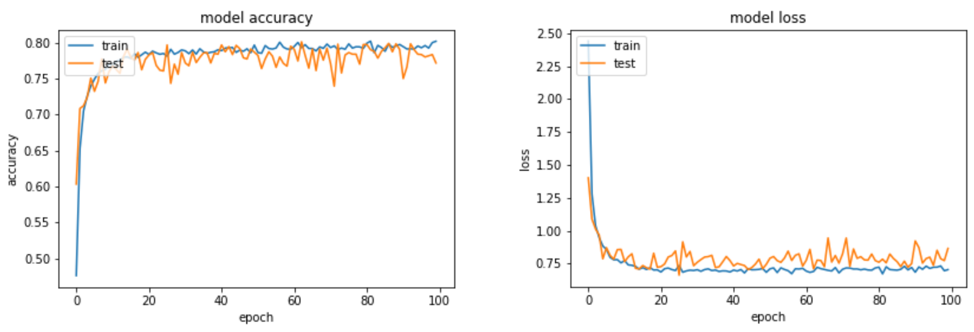


Figure 9.8. VGG-16 Accuracy and loss curve.

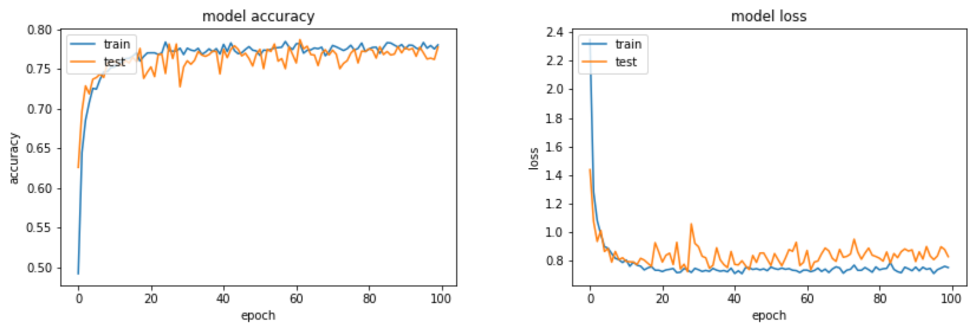


Figure 9.9. VGG-19 Accuracy and loss curve.

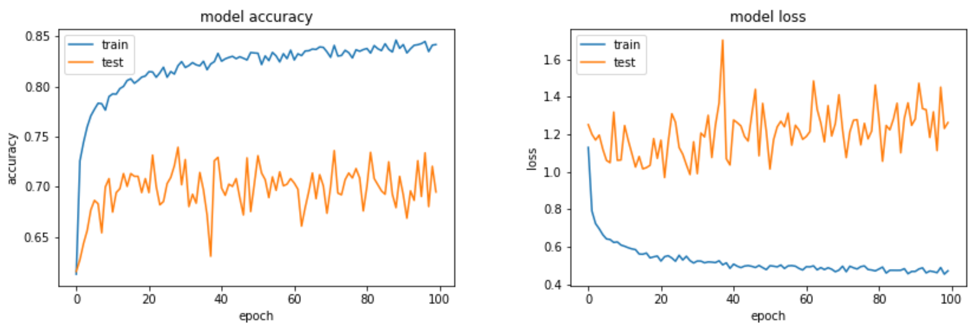


Figure 9.10. ResNet50 Accuracy and loss curve.

PART 10

CONCLUSION

The study in the thesis consists of two phases. In the first phase, we studied on lab-based butterfly images taken on under a fixed protocol. The species of butterflies in these images are identified by expert entomologists. We used a total of 140 images for lab-based butterfly images of 10 species. After applying some preprocess to the images such as histogram equalization and background removing, we extracted several features from the butterfly images. Finally, we used an artificial neural network in MATLAB version R2014b using the Neural Network Toolbox for butterfly identification. The ANN model achieved an accuracy of 98%.

In the second phase, a field-based data set was created using butterfly images which are classified by expert entomologists taken from nature. The input images of butterflies are classified with deep learning architectures without using any feature extraction method. Transfer learning was carried out using pre-trained models. Comparison and evaluation of the experimental results obtained using three different network structures are conducted. According to the results, the highest success was achieved by VGG16 architecture. Although the images have some problems such as the position of butterflies, the shooting angle, butterfly distance, occlusion and background complexity, 80% success was achieved for both test and training data. In conclusion, we observed that the transfer learning approach can be successfully applied in nature images. Our future research includes employing a mobile application by using the proposed pre-trained model in this study.

Figure 10 show the summary of the computational tests conducted on two stages in the thesis.

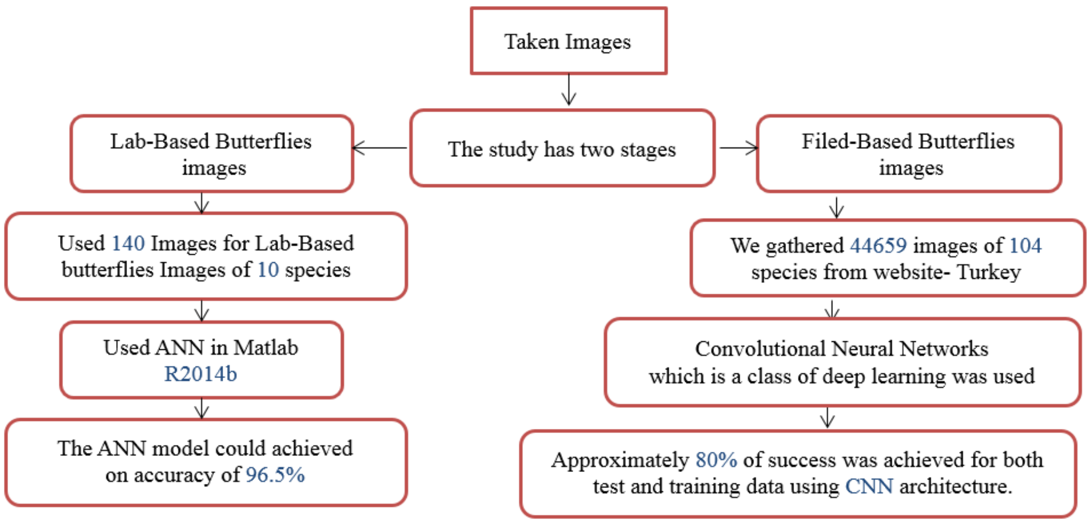

Taken Images
The study has two stages
Lab-Based Butterflies images
Filed-Based Butterflies images
Used 140 Images for Lab-Based butterflies Images of 10 species
We gathered 44659 images of 104 species from website- Turkey
Used ANN in Matlab R2014b
Convolutional Neural Networks which is a class of deep learning was used
The ANN model could achieved on accuracy of 96.5%
Approximately 80% of success was achieved for both test and training data using CNN architecture.

Figure 10.1. Summary of the computational tests.

REFERENCES

1. Stork, N. E., "How Many Species of Insects and Other Terrestrial Arthropods Are There on Earth?" ***Annual review of entomology***, 63: 31-45 (2018).

2. Pinzari M., Santonico, M., Pennazza, M. et al., "Chemically mediated species recognition in two sympatric Grayling butterflies: Hipparchia fagi and Hipparchia Hermione (Lepidoptera: Nymphalidae, Satyrinae)", ***PloS ONE Accelerating the publication of peer-reviewed science***, 13(6): 1-14 (2018).

3. Kaya, Y., Ertuğrul, O. F, Tekin, R., "Two novel local binary pattern descriptors for texture analysis", ***Applied Soft Computing***, 34: 728-735 (2015).

4. Kaya, Y., Kaycı, L., "Application of artificial neural network for automatic detection of butterfly species using color and texture features", ***The Visual Computer***, 30: 71-79 (2014).

5. Wen, C., Guyer, D., "Image-based orchard insect automated identification and classification method", ***Computers and Electronics in Agriculture***, 89: 110-115 (2012).

6. Xie, C., Zhang, R., Li, R., Li, J., Hong, P., Xia, J., Chen, P., "Automatic classification for field crop insects via multiple-task sparse representation and multiple-kernel learning", ***Computers, and Electronics in Agriculture***. 119: 123-132 (2015).

7. Feng, L., Bhanu, B., Heraty, J., "A software system for automated identification and retrieval of moth images based on wing attributes", ***Pattern Recognition*** 51: 225-241 (2016).

8. Yao, Q., Lv. J., Liu, Q.-J., Diao, G-Q., Yang, B-J., Chen, H-M., Tang, J., "An insect imaging system to automate rice light-trap pest identification", ***Journal of Integrative Agriculture***, 11: 978-985 (2012).

9. Faithpraise, F., Birch, P., Young, R., Obu, J., Faithpraise, B., Chatwin, C., "Automatic plant pest detection and recognition using k-means clustering algorithm and correspondence filters", ***International Journal of Advanced Biotechnology and Research***, 4(2): 189-199 (2013).

10. Leow, L. K., Chew, L-L., Chong, V. C., Dhillon, S. K., "Automated identification of copepods using digital image processing and artificial neural network", ***BMC Bioinformatics***, 16: S4 (2015).

11. Zhu L.-q., Zhang, Z., " Insect recognition based on integrated region matching and dual-tree complex wavelet transform", ***Journal of Zhejiang University SCIENCE C***, 12: 44-53 (2011).

12. Mayo, M., Watson, A. T., "Automatic species identification of live moths", ***Knowledge-Based Systems,*** 20 (2): 195-202 (2007).

13. Watson, A. T., O'Neill, M. A., Kitching, I. J., "Automated identification of live moths (Macrolepidoptera) using digital automated identification System (DAISY)", ***Systematics and Biodiversity***, 1 (3): 287-300 (2004).

14. Silva, F. L. D., Grassi, S. M. L., Francoy, T. M., Costa, A. H. R., "Evaluating classification and feature selection techniques for honeybee subspecies identification using wing images", ***Computers and Electronics in Agriculture***, 114: 68-77 (2015).

15. Wang, J., Lin C., Ji, L., Liang, A., "A new automatic identification system of insect images at the order level", ***Knowledge-Based Systems,*** 33: 102-110 (2012).

16. Buschbacher, K., Ahrensb, D., Espeland, M., Steinhagea, V., "Image-based species identification of wild bees using convolutional neural networks", ***Ecological Informatics***, 55: 101017 (2020).

17. Feng, L., Bhanu, B., Heraty, J., "Identification and Retrieval of Moth Images Based on Wing Patterns", ***Springer,*** pp 22: 349-369 (2015).

18. Angelo, C., Guiam, Miguel. J, Bawagan, J., "Insectify: An Android Application for Digital Insect Identification Using A Convolutional Neural Network", Institute ***of Computer Science***, 190: 1-10 (2017)

19. Xue, A., Li, F., Xiong, Y., "Automatic Identification of Butterfly Species Based on Gray-Level Co-occurrence Matrix Features of Image Block", ***Journal of Shanghai Jiaotong University (Science)***, 24: 220–225 (2019).

20. Internet: Adamerkelebek, "Butterflies Monitoring & Photography Society of Turkey", **http://adamerkelebek.org**, (2019).

21. Wang, J., Ji, L., Liang, A, Yuan, D., "The identification of buttery families using content-based image retrieval", ***Biosystems Engineering***, 111 (1): 24-32 (2012).

22. O'Neill, M. A., "DAISY: Automated Taxon Identification in Systematics: Theory, Approaches, and Applications", ed chap.9, ***MacLeod, Natural History Museum London, UK,*** 131-152, (2008).

23. Dietrich, C. H., Pooley, C. D., "Automated identification of leafhoppers (Homoptera: Cicadellidae: Draeculacephala Ball)", ***Annals of the Entomological Society of America***, 87 (4): 412-423 (1994).

24. Dietrich. C. H, Emigh. T. H, Deitz. L. L. "Morphometric discrimination among females of sibling species of Aconophorini (Homoptera: Membracidae)", ***Systematic entomology***, 16 (3): 311-31 (1991).

25. Solis-Sanchez, L. O, Garca-Escalante, J. J., Castaneda-Miranda, R., Torres-Pacheco, I., Guevara-Gonzalez, R., "Machine vision algorithm for whiteies

(Bemisia tabaci Genn.) scouting under greenhouse environment", ***Journal of Applied Entomology***, 133 (7): 546-552 (2009).

26. Zhu, L-Q., Zhang, Z., "Auto-classification of insect images based on color histogram and GLCM", in Fuzzy Systems and Knowledge Discovery (FSKD), ***Seventh International Conference on IEEE***, ***Yantai, China,*** 6: 2589-2593 (2010).

27. Wen, C., Zhu, Q., "Dimension reduction analysis in image-based species classification, in Intelligent Computing and Intelligent Systems (ICIS)", International ***Conference on IEEE***, ***Yantai, China***, 3: 731-734 (2010).

28. Yalcin H., "Vision-based automatic inspection of insects in pheromone traps", in Agro-Geoinformatics (Agro-geoinformatics), ***Fourth International Conference on, IEEE, Istanbul, Turkey***, 10: 333-338 (2015).

29. Arbuckle, T., Schroder, S., Steinhage, V., Wittmann, D., "Biodiversity informatics in action: identication and monitoring of bee species using ABIS", in Proc. ***15th Int. Symp. Informatics for Environmental Protection, Zürich,*** 1: 425-430 (2001).

30. Nanni, L., Lumini, A., Brahnam S., "Survey on LBP based texture descriptors for image classification", ***Expert Systems with Applications***, 39: 3634–3641 (2012).

31. Pietikäinen, M., Ojala, T., "Texture Analysis in Industrial Applications". In: Sanz J.L.C. (eds) Image Technology. ***Springer, Berlin, Heidelberg, Sanz, Jorge L.C***. 337-359 (1996).

32. Ojala, T., Pietikainen, M., and Maenpaa, M., "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns". ***IEEE Transaction on Pattern Analysis and Machine Intelligence***, 24: 971–987 (2002).

33. Heikkila, M., Pietikainen, M., & Schmid, C., "Description of interest regions with local binary patterns". ***Pattern Recognition***, 42, 425–436 (2009).

34. Martineau, M., Conte, D., Raveaux, R., Arnault, I., Munier, D., Venturini, G., "A survey on image- based insect classification". ***Pattern Recognition***, 65: 273-284 (2017).

35. Al-Saqer, S. M., Hassan, G. M., "Artificial neural networks based red palm weevil (Rynchophorus Ferrugineous, Olivier) recognition system", ***Am. J. Agric. Biol. Sci***, 6: 356-364 (2011).

36. To_lski, A., "DrawWing, a program for numerical description of insect wings", ***Journal of Insect Science***, 4 (1): 17 (2004).

37. Ojala, T., Pietik¨ainen, M., & Harwood, D., "A comparative study of texture measures with classification based on featured distributions". ***Pattern Recognition***, 29: 51–59 (1996).

38. LeCun, Y., Bengio, Y., & Hinton, G., "Deep learning", ***Springer nature***, 521: 436-444 (2015).

39. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z., "Rethinking the inception architecture for computer vision". ***In Proceedings of the IEEE conference on computer vision and pattern recognition***, pp. 2818-2826 (2016).

40. Collobert, R., & Weston, J., "A unified architecture for natural language processing: Deep neural networks with multitask learning". ***In Proceedings of the 25th international conference on Machine learning,*** pp. 160-167(2008).

41. Hinton, G. et al., "Deep neural networks for acoustic modeling in speech recognition". ***IEEE Signal processing magazine***, 29: 82-97 (2012).

42. LeCun, Y., & Bengio, Y., "Convolutional networks for images, speech, and time series". ***The handbook of brain theory and neural networks", Cambridge, MA, USA*** 10: 255–258 (1998).

43. Wang, J., Yang, Y., Mao, J., Huang, Z., Huang, C., & Xu, W., “Cnn-rnn: A unified framework for multi-label image classification". ***In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Las Vegas, NV, USA***, 2285-2294 (2016).

44. Schroff, F., Kalenichenko, D., & Philbin, J., “Facenet: A unified embedding for face recognition and clustering". ***In Proceedings of the IEEE conference on computer vision and pattern recognition, Boston, MA, USA***, pp. 815-823 (2015).

45. Cheng, D., Gong, Y., Zhou, S., Wang, J., & Zheng, N., "Person re-identification by multi-channel parts-based cnn with improved triplet loss function". ***In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA***, pp. 1335-1344 (2016).

46. Sermanet, P., & LeCun, Y., "Traffic sign recognition with multi-scale Convolutional Networks". ***In The International Joint Conference on Neural Networks (IJCNN), San Jose, CA, USA***, pp. 2809-2813 (2011).

47. Pereira, S., Pinto, A., Alves, V., & Silva, C. A., "Brain tumor segmentation using convolutional neural networks in MRI images". ***IEEE transactions on medical imaging, United States National Library of Medicine***, 35(5): 1240-1251 (2016).

48. Liang, M., & Hu, X., "Recurrent convolutional neural network for object recognition". ***In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Boston, MA, USA***, pp. 3367-3375 (2015).

49. Chollet, F., "Deep Learning mit Python und Keras: Das Praxis-Handbuch vom Entwickler der Keras-Bibliothek". ***MITP-Verlags GmbH & Co***. KG 978-3-95845-840-6 (2018).

50. Krizhevsky, A., Sutskever, I. and Hinton, G. E., "Imagenet classification with deep convolutional neural networks". ***In Advances in neural information processing systems***, pp. 1097–1105 (2012).

51. Simonyan, K., & Zisserman, A., "Very deep convolutional networks for large-scale image recognition". ***Cornell university***, ***arXiv preprint arXiv***: 1409.1556 (2014).

52. Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., & Adam, H., "Mobilenets: Efficient convolutional neural networks for mobile vision applications". ***Cornell university arXiv preprint arXiv***: 1704.04861 (2017).

53. He, K., Zhang, X., Ren, S., & Sun, J., "Deep residual learning for image recognition". ***In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), Las Vegas, NV, USA***, pp. 770-778 (2016).

54. Deng, J., et al. "Imagenet: A large-scale hierarchical image database", ***IEEE conference on computer vision and pattern recognition. Miami, FL, USA***, pp: 248-255 (2009).

55. Canziani, A., Paszke, A. and Culurciello, E., "An analysis of deep neural network models for practical applications". ***Cornell university, arXiv preprint arXiv***:1605.07678 (2016).

56. Seung-Ho, K., Tae-Hee, K., "A Performance Improvement of Automatic Butterfly Identification Method Using Color Intensity Entropy", ***The Journal of the Korea Contents Association***, 17(5): 624-632 (2017).

57. Li, F., Xiong, Y., "Automatic identification of butterfly species based on HoMSC and GLCMoIB", ***Vis Comput Springer-Verlag*** 34: 1525–1533 (2018)

58. Kalafi, Y., Town. C., Dhillon, S. K., "How automated image analysis techniques help scientists in species identification and classification", ***Folia Morphologica*** 77(2): 179- 193 (2018).

59. Russell, K. N., Do, M. T., Platnick, N. I., "Introducing SPIDA-Web: an automated Identification system for biological species". ***In: Proceedings of Taxonomic Database Working Group Annual Meeting***, pp.11–18 (2005).

60. Kaya, Y., Kayci, L., "Application of artificial neural network for automatic detection of butterfly species using color and texture features", ***The Visual Computer***, 30: 71–79 (2014).

61. Yang, J., Yan, X., Yao, B., "Character Feature Extraction Method based on Integrated Neural Network", ***AASRI Procedia, Identification and Control***. 3: 197-202 (2012).

RESUME

Ayad saad ALMRYAD was born in Algmail -Libya in 1968 and he graduated first and elementary education in this city. He completed high school education in Algmail High School, after that, he started the high diploma program at the Higher Institute of Industrial Technology - Tripoli Department of Electricity in 1987. Then in 1995, he started his assignment as a Research Assistant in the higher Institute of Technology - Regdaleen Department of Electrical Engineering, from 1997 until 2006 A Head of Electricity Department. To complete M. Sc. education, he moved to University Utara Malaysia (UUM) Department Computer science, Information Communication Technology (I C T), and from 2008 until 2014 A Head of Computer Department. Where he has been still working as a H.C.D for.

CONTACT INFORMATION

Address: Karabük University
Graduate School of Natural & Applied Science
Demir-Çelik Campus / KARABUK
E-mail: yd_saad2010@yahoo.com