ART AUTHENTICATION USING DCNN

By

DEVI K.R (13408036)

of

UNIVERSITY OF KERALA

A PROJECT REPORT

Submitted to the

FACULTY OF COMPUTER SCIENCE AND ENGINEERING

In partial fulfillment of the requirements

for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

April, 2017

MAR BASELIOS COLLEGE OF ENGINEERING & TECHNOLOGY

Bethany Hills, Nalanchira

Thiruvananthapuram 15

UNIVERSITY OF KERALA

MAR BASELIOS COLLEGE OF ENGINEERING & TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Nalanchira, Thiruvananthapuram.

CERTIFICATE

This is to certify that the work reported in the project entitled ***Art Authentication using DCNN*** *is the bonafide record of work done by* ***Devi K.R (REG NO:13408036), Samuel Punnoose John (REG NO:13408090) & Sharen Augustine (REG NO:13408095)*** *of the* ***DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING*** *during the year 2016-2017 in partial fulfillment for the award of the degree of* ***Bachelor of Technology*** *by the UNIVERSITY OF KERALA.*

Project Coordinator   Internal Guide   Head of the Department

Submitted for the project Viva Voce examination held on ....

Internal Examiner   External Examiner

Acknowledgement

First and foremost, we wish to thank Fr. Wilson Thattaruthundil, Bursar, Mar Baselios College of Engineering and Technology for his support and prayers. We wish to thank Dr. T. M. George, Principal, Mar Baselios College of Engineering and Technology for encouraging and leading us through this endeavour. We also wish to convey our heartful gratitude to Ms. Tessy Mathew Head of Department, Computer Science & Engineering, Mar Baselios College of Engineering and Technology. We would also like to place on records our ardent and earnest gratitude to our project guide Mr. Sandeep S. R, Assistant Professor, Department of Computer Science and Engineering, Mar Baselios College of Engineering and Technology. We are extremely happy to mention a great word of gratitude to the entire faculty, Department of Computer Science and Engineering for providing us with all facilities for the completion of this work.

ABSTRACT

The objective of this initiative is to help detect paintings done by an artist using deep convolutional neural network. The image is filtered through thousands of neurons with millions of connections to extract content patterns,style patterns and features of paintings done by an artist. These are then combined together context-sensitively using optimization algorithms that find the best ways to combine everything together(extract features). We examine the set of paintings and determine the paintings that were painted by the same artist. The training set and test set consists of artwork images and their corresponding class labels (painters).

Contents

List of Figures

Chapter 1

INTRODUCTION

Image classification is the task of taking an input image and outputting a class or a probability of classes that best describes the image. For humans, this task of recognition is one of the first skills we learn from the moment we are born and is one that comes naturally and effortlessly as adults. Without even thinking twice, were able to quickly and seamlessly identify the environment we are in as well as the objects that surround us. When we see an image or just when we look at the world around us, most of the time we are able to immediately characterize the scene and give each object a label, all without even consciously noticing.In this project we tend to classify paintings on the basis of their respective painters by using a class of biologically inspired vision model called Deep Convolutional Neural Network which was inspired from the above mentioned human performance.

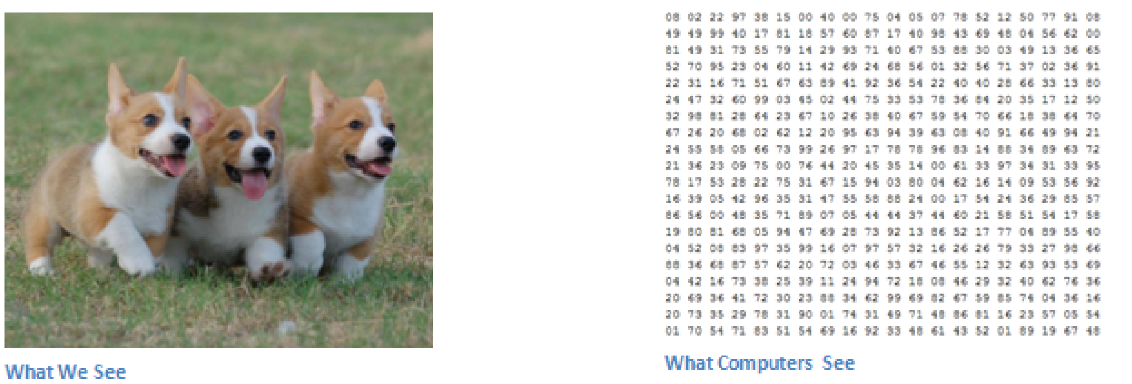

Figure 1.1: Representation of picture as matrix

1.1 OBJECTIVES

To detect whether a given painting is painted by one of the painters in the dataset by passing it through a series of convolutional layers,non linear,pooling and fully connected layers. The purpose of this project is to prevent forgery in the world of art.

1.2 SCOPE OF THE PROJECT

In recent years, DCNN has been used extensively in a wide range of fields such as computer vision,natural language processing, image processing etc. In deep learning, Convolutional Neural Networks are found to give the most accurate results in solving real world problems. DCNNs achieve better classification accuracy on large scale datasets due to their capability of joint feature and classifier learning. DCNN in image processing is a potential trend for the development of image recognition. The main scope is in the field of art.

Chapter 2

LITERATURE SURVEY

2.1 A Neural Algorithm of Artistic Style

This paper introduces an artificial system based on a Deep Neural Network that creates artistic images of high perceptual quality. The system uses neural representations to separate and recombine content and style of arbitrary images, providing a neural algorithm for the creation of artistic images.

2.2 Stochastic Pooling for Regularization of Deep Convolutional Neural Networks

This paper introduces a simple and effective method for regularizing large convolutional neural networks. Replaces the conventional deterministic pooling operations with a stochastic procedure, randomly picking the activation within each pooling region according to a multinomial distribution, given by the activities within the pooling region.

Chapter 3

IMPLEMENTATION METHODOLOGY

3.1 PROBLEM STATEMENT

To examine a given painting and determine its artist by passing unique datasets of art as input.

3.2 PROBLEM DESCRIPTION

To identify an authentic work of art from forgery by using DCNN and computer vision skills that engage with a unique dataset of art. We create an algorithmic understanding of how humans create and perceive artistic forgery.

Morellian analysis is based on the creation and mapping of formulae describing repeated stylistic details in the artwork and reflecting the particular approach of the artist. This form of authentication relies on keen eyes of art historians who use their knowledge of the uniqueness and the progression of the artist's style to conclude whether a piece of art is authentic or not.Technical analysis utilizes equipment such as microscopes to view the oxidized cracks of oil paintings,or the extra layers on ancient glasses.This technique is used to see if there are parts of the artwork that have been artificially induced.Since there are no standard methods to find image forgery,we use deep convolution neural networks to process visual images.

3.3 FEATURES OF THE PROJECT

To identify an authentic work of art using deep convolutional neural network. An artificial system is introduced based on deep convolution neural network that detect forgery of artistic images of high perceptual quality. Requirements of an user are to test an image and identify whether the painting is original or not. Requirements of the developer are unique datasets of images by different painters as training set, convert datasets to predefined dimensions,create predictive models using unsupervised machine learning,accept input given by user in a compatible format and predict if the image is original or not.

3.4 PROBLEM SOLVING METHOD

3.4.1 PROPOSED SYSTEM

A deep convolutional neural network is used here that process visual informations hierarchially.For image synthesis average pooling operation is used which improves gradient flow and more appealing results are produced.To visualize images gradient descent is used.

3.5 UML DIAGRAMS

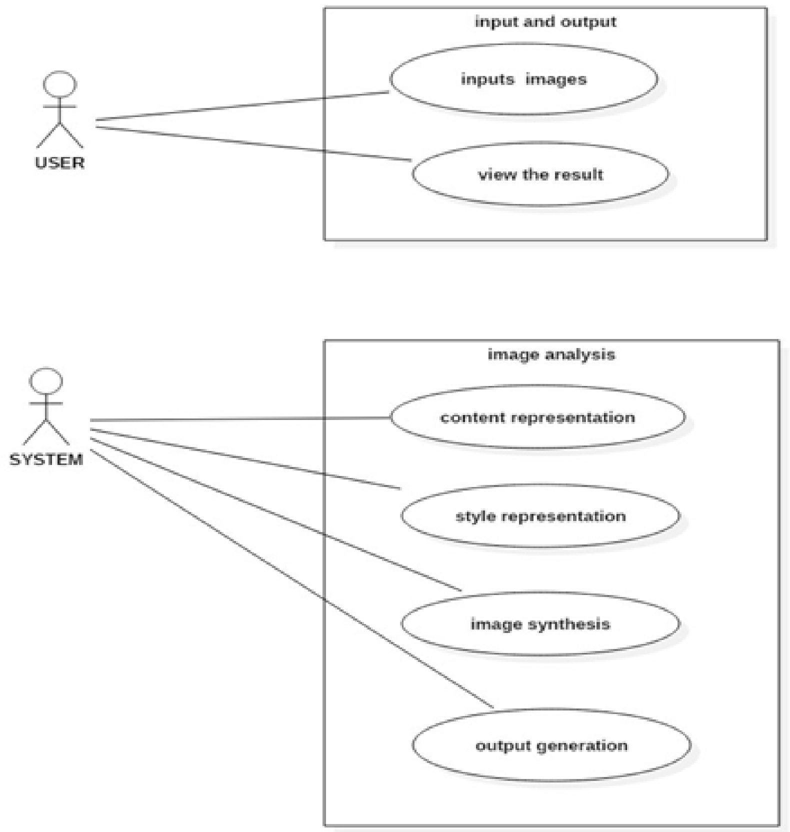

Figure 3.1: Use Case Diagram

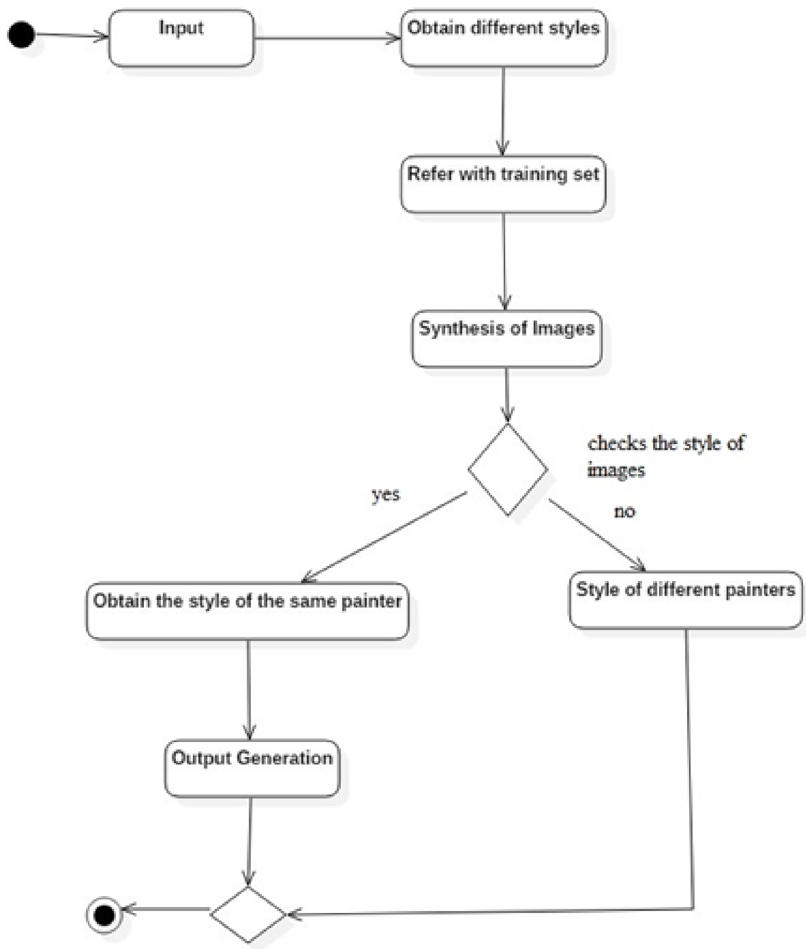

Figure 3.2: Activity Diagram

Chapter 4

PSEUDO CODE

4.1 NETWORK CONFIGURATIONS

To configure a network you must create a config file. Config file describes the network structure, training parameters and all other possible configuration.

4.2 LAYER SPECIFICATION

Describes the network structure and input/output size. Example:

4.3 TRAINING A NETWORK

First create a network using a config file

```
%%%%%%%%%%%%%%%%%%%%%  input settings  %%%%%%%%%%%%%%%%%%

       net.hyperParam.sizeFmInput = [28 28];   % size of input feature map
       net.hyperParam.numFmInput  = 1;         % number of feature map of the input

%%%%%%%%%%%%%%%%%%%%%  Layers specification %%%%%%%%%%%%%%%%%%

       net.layers{end+1}.properties = struct('type',2,'numFm',4  ,'kernel',4,'pad',2);
       net.layers{end+1}.properties = struct('type',2,'numFm',5 ,'kernel',6);
       net.layers{end+1}.properties = struct('type',1,'numFm',60);
       net.layers{end+1}.properties = struct('type',1,'numFm',10);

%%%%%%%%%%%%%%%%%%%%%  Hyper params - training %%%%%%%%%%%%%%%%%%

       net.hyperParam.trainLoopCount = 10000;         %on how many images to train before evaluating the network
       net.hyperParam.testImageNum   = 20000;         % after each loop, on how many images to evaluate network performance
       net.hyperParam.ni_initial     = 0.05;          % ni to start training process
       net.hyperParam.ni_final       = 0.0025;        % final ni to stop the training process
```

Figure 4.1: Network Configuration

```
%%%%%%%%%%%%%%%%%%%%%%  input settings  %%%%%%%%%%%%%%%%%%

net.hyperParam.sizeFmInput = [32 32];   % size of input feature map
net.hyperParam.numFmInput  = 3;         % number of feature map of the input

%%%%%%%%%%%%%%%%%%%%%%  Layers specification %%%%%%%%%%%%%%%%%%

net.layers{end+1}.properties = struct('type',2, 'numFm',48 , 'Activation',@Relu, 'dActivation',@dRelu, 'kernel',5, 'stride',2);
net.layers{end+1}.properties = struct('type',2, 'numFm',48 , 'Activation',@Relu, 'dActivation',@dRelu, 'kernel',5, 'pad',1, 'dropOut',0.8);
net.layers{end+1}.properties = struct('type',2, 'numFm',96 , 'Activation',@Relu, 'dActivation',@dRelu, 'kernel',7, 'pad',2, 'dropOut',0.8);
net.layers{end+1}.properties = struct('type',1, 'numFm',128, 'Activation',@Relu, 'dActivation',@dRelu, 'dropOut',0.8);
net.layers{end+1}.properties = struct('type',1, 'numFm',10);

%%%%%%%%%%%%%%%%%%%%%%  Hyper params - training %%%%%%%%%%%%%%%%%%

net.hyperParam.trainLoopCount = 1000;  %on how many images to train before evaluating the network
net.hyperParam.testImageNum   = 2000;   % after each loop, on how many images to evaluate network performance
net.hyperParam.ni_initial     = 0.0001;  % ni to start training process
net.hyperParam.ni_final       = 0.000001; % final ni to stop the training process
net.hyperParam.momentum       = 0.9;
net.hyperParam.lambda         = 0.008;  % L2 regularization factor
net.hyperParam.flipImage      = 1; % randomly flip the input hor/vert before passing to the network. Improves learning in some instances
net.runInfoParam.verifyBP     = 0;
```

Figure 4.2: Layer Specification

net = CreateNet('../../Configs/mnist.conf');

Then, call Train function with the dataset containing the train/test samples: net = Train(MNIST,net, 15000);

Here , MNIST is the dataset , this will train for 15000 images from the test set in a cyclic manner. In order to train longer , you can specify Inf as the last parameter, network will train until learning rate (ni) reach below the given threshold.

Chapter 5

SOFTWARE DESCRIPTION

The training set is unbalanced and some classes are only present in the training set and some only in the test set. Additionally input images are of various dimensions. There are 79433 instances and 1584 unique painters in the training set and the test set is composed of 23817 instances.There are a total of 1584 painters in the training set.

The model assumes fixed-size inputs, so the first preprocessing step was to resize each image's smallest dimension to 28 pixels (retaining the aspect ratio) and then cropping it at the center of the larger dimension, obtaining 28x28 images. Some information gets lost during this process and an alternative approach where multiple crops are taken from the same image was not considered. During the training phase random transformations (rotations, zooms, shifts, shears and flips) were applied to data.

Each of these of these images are then passed onto the convolution and pooliing layers of the DCNN.

The output is then passed onto a feature classifier to classifier each painting to their respective painter based on the unique features of each painting in kernel which was obtained by using unsupervised machine learning.

Chapter 6

LIMITATIONS OF THE PROJECT

1. Computation becomes difficult as the number of paintings increase.
2. Time taken during execution of code increases as the number of paintings increase.
3. All the paintings of a painter may not be detected accurately.

Chapter 7

FUTURE ENHANCEMENTS

- Use machine with higher processing capacity.
- Use platform/language with lesser overhead and more built-in functions.

Chapter 8

Conclusion

We introduced an artificial system based on a Deep Convolutional Neural Network that identifies artistic images of high perceptual quality based on their unique painters. The system uses neural representations providing a neural algorithm to train and test artistic images based on it's painter.Our work offers a path forward to an algorithmic understanding of image recognition and feature extraction of artwork using DCNN and unsupervised machine learning.

Chapter 9

APPENDIX-I:WORKING ENVIRONMENT

9.1 SOFTWARE SPECIFICATION

1. Latest version of MATLAB with all libraries including that for neural networks and computer vision.

2. Data sets of training and test images along with their labels.

9.2 HARDWARE SPECIFICATION

1. A computer with 8GB RAM and an hard disk with capacity of 100GB.

2. Processor should have dedicated VGA card with a octa core GPU.

Chapter 10

APPENDIX-II:SCREENSHOTS

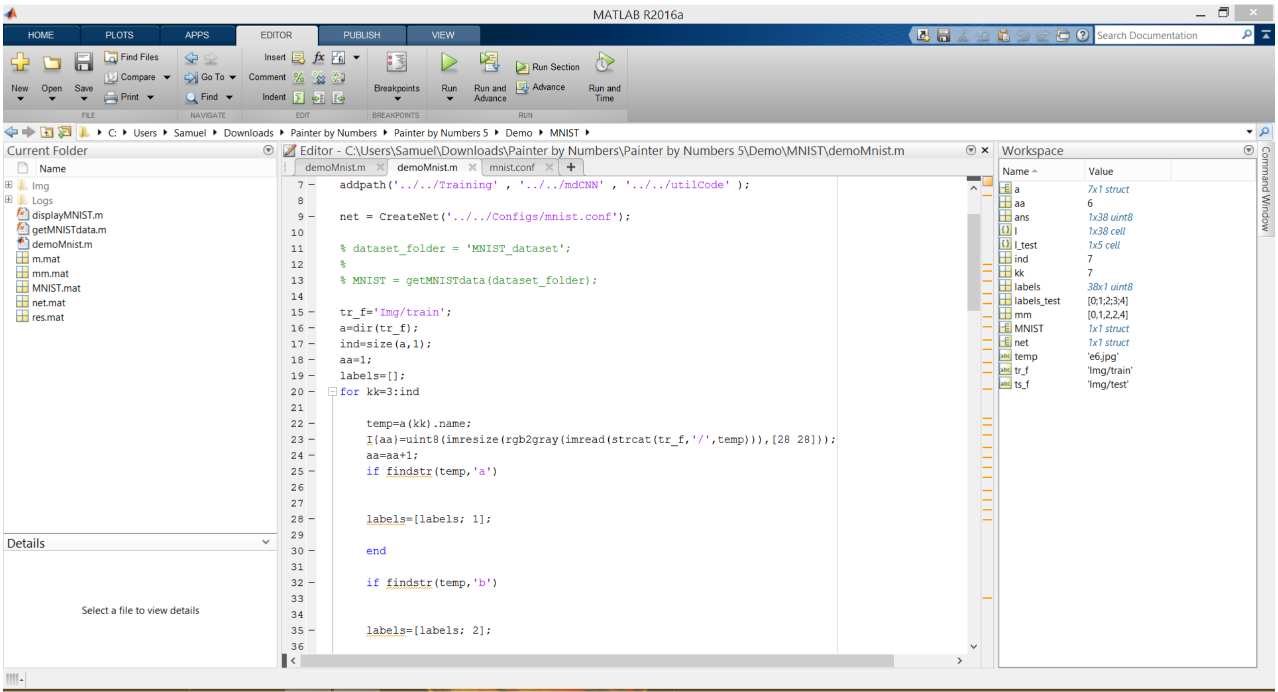

Figure 10.1: Screenshot of program I

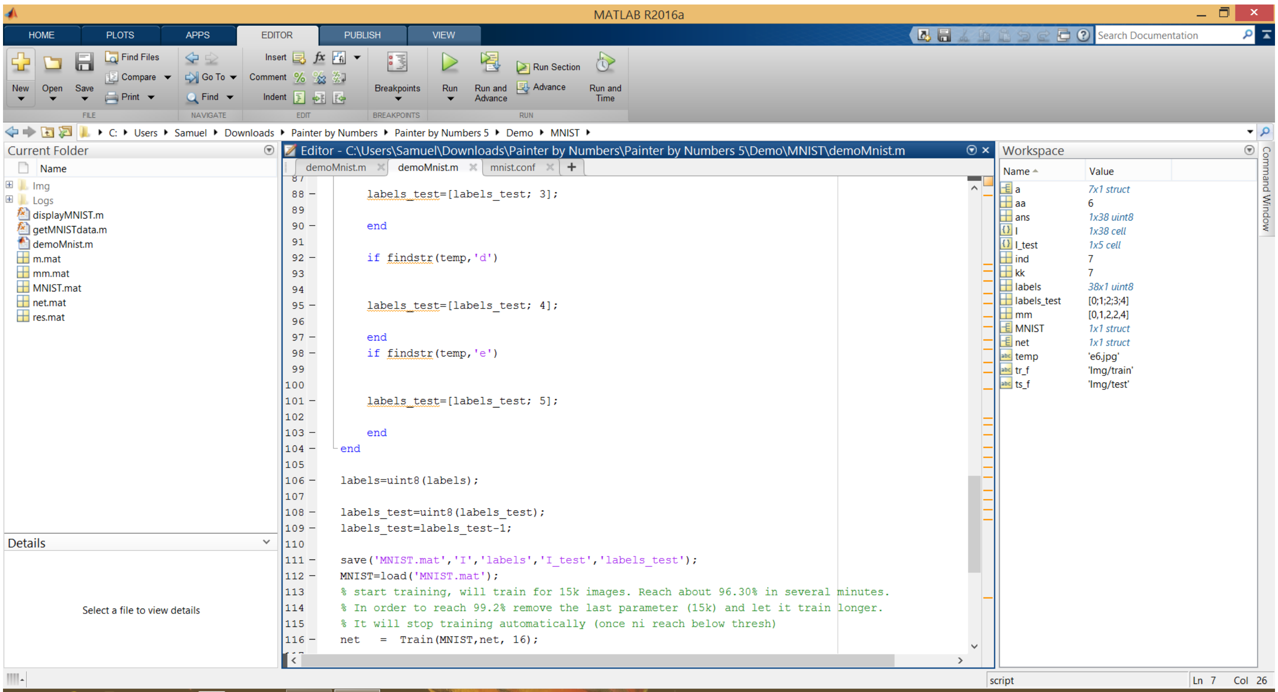

Figure 10.2: Screenshot of program II

```
Start training iterations
Iter 1   | Imgs=10000 | time=190.84 | TrainErr=0.039739 | meanGrad=0.010557 | meanWeight=0.023438 | varWe
Finish training. max samples reached
Testing on 5 images...
success rate 80.000000%
Detected Output is :
     0     1     2     2     4
```

Figure 10.3: Output

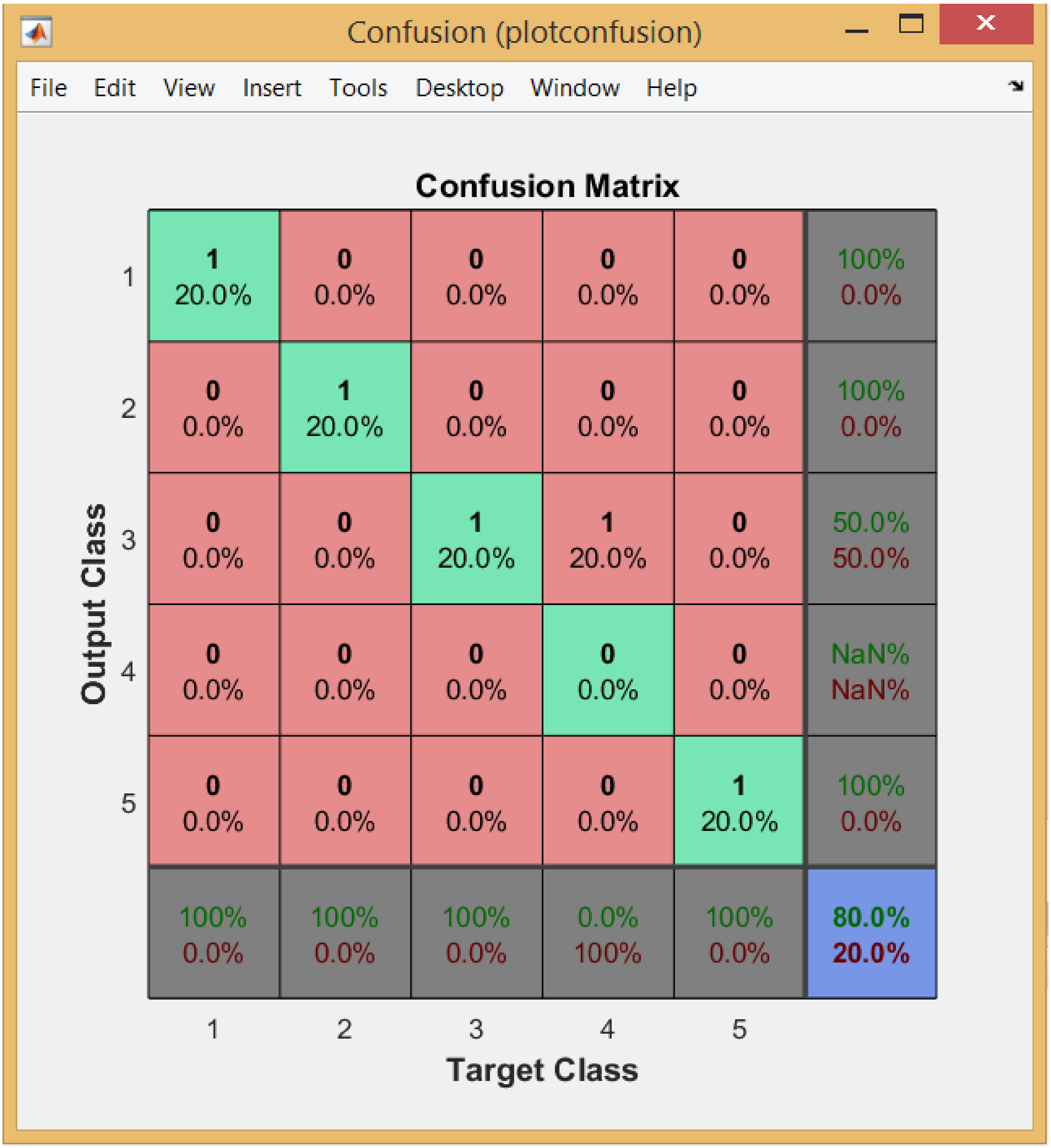

Figure 10.4: Confusion Matrix Representation Of Output

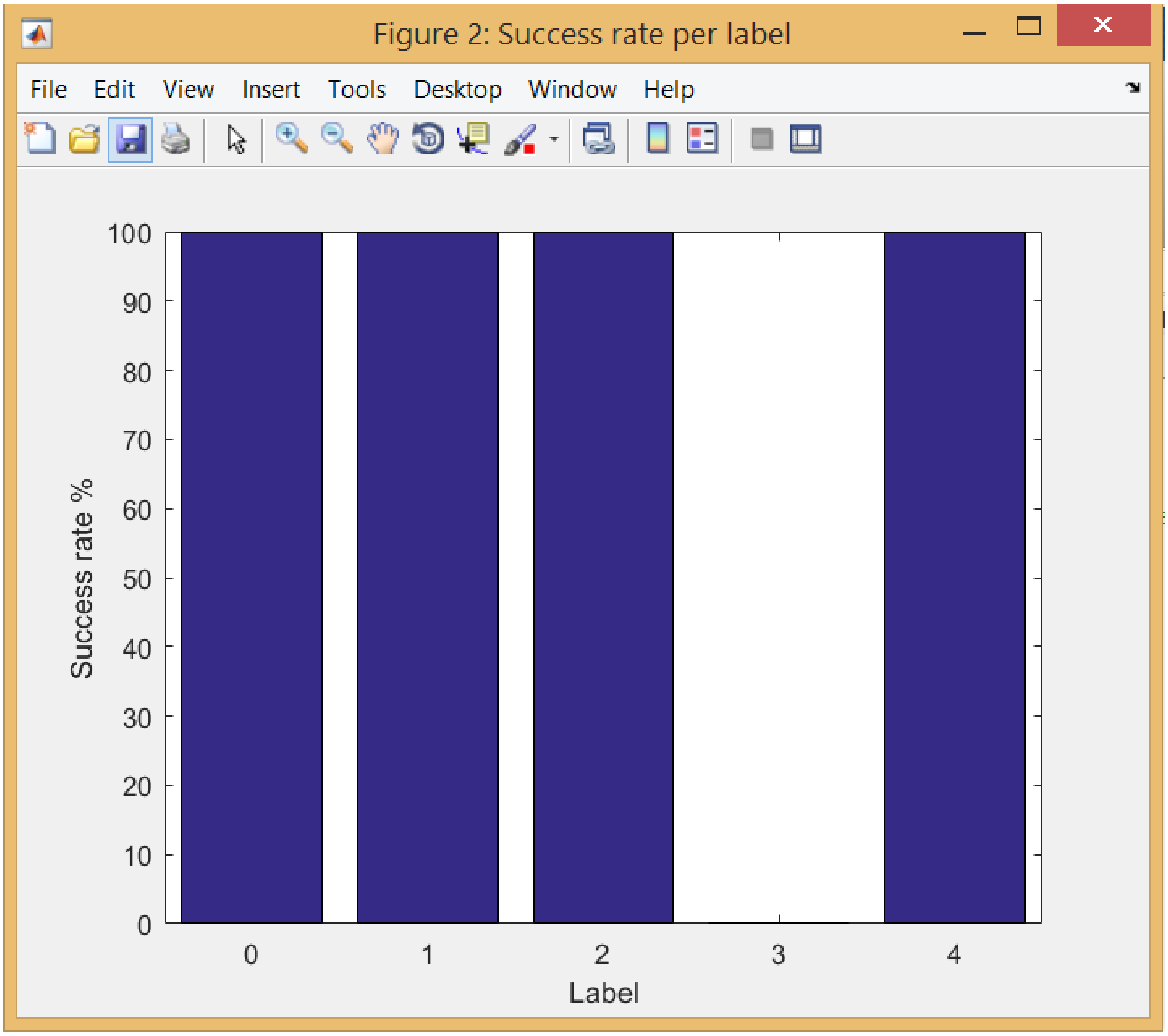

Figure 10.5: Bar Graph Representation Of Output

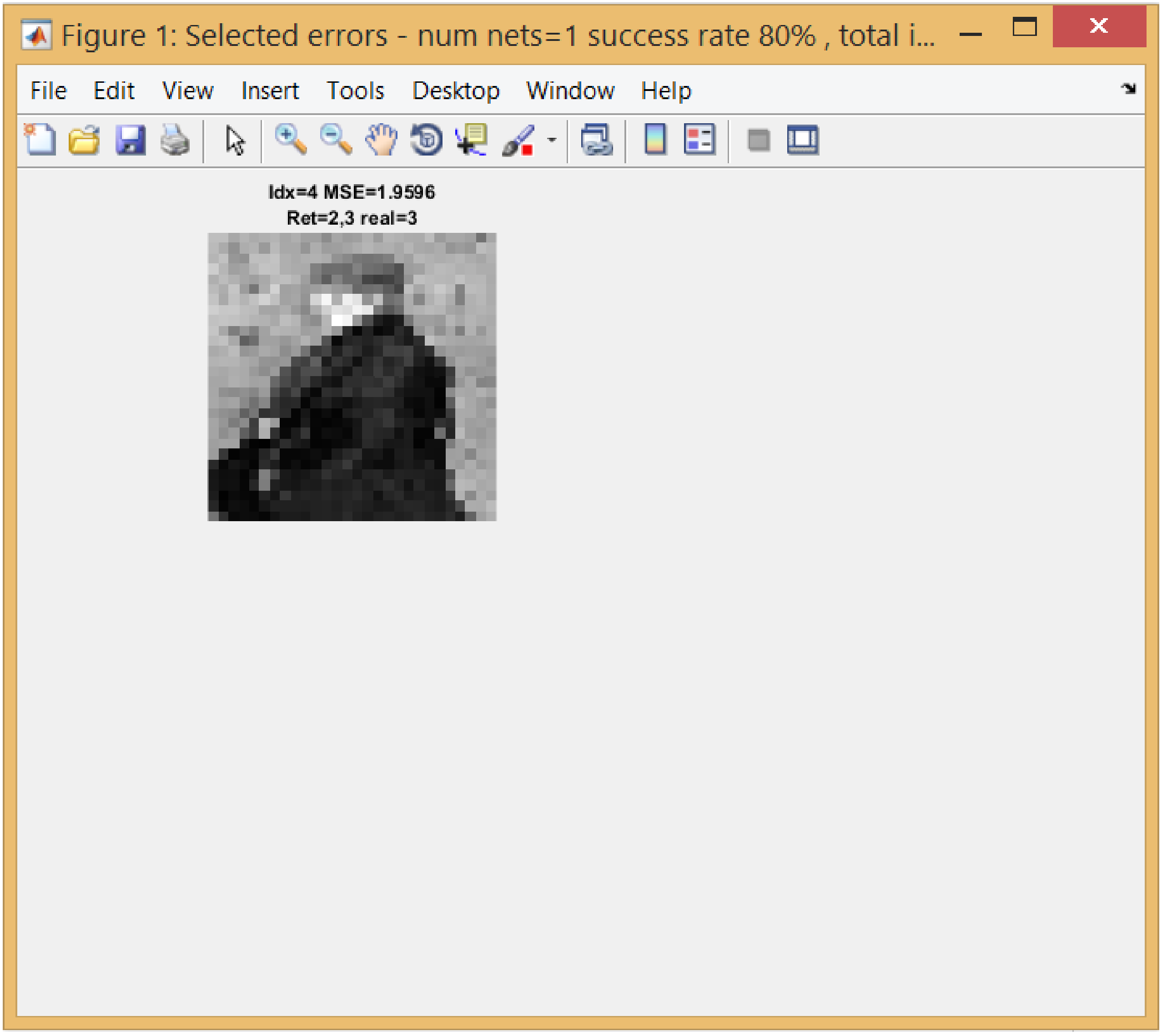

Figure 10.6: Wrongly Detected Picture

REFERENCES

[1] Gatys, Leon A., Alexander S. Ecker, and Matthias Bethge. ”A neural algorithm of artistic style.” arXiv preprint arXiv:1508.06576 (2015).

[2] Zeiler, Matthew D., and Rob Fergus. ”Stochastic pooling for regularization of deep convolutional neural networks.” arXiv preprint arXiv:1301.3557 (2013).