TomatoX: Vision-based Judgment of Tomato Maturity using Extreme Learning Machines

|  |
| --- |
| **Roberto Saldaña Jr.**  **Division of Natural Sciences and Mathematics, University of the Philippines Visayas Tacloban College**  **Magsaysay Boulevard, Tacloban City, Leyte 6500**  ***roberto.saldana.jr@gmail.com*** |

Abstract

Tomato is one of the economically important fruit in the world. Major supplies of tomatoes come from farms and greenhouses. The traditional way of sorting tomatoes are done by human graders. However, this procedure has a lot of shortcomings. Manual sorting is a very time-consuming, tedious, and expensive task. Additionally, the human identification of colors is affected by many factors like brightness, intensity, etc. A lot of research has been conducted on the automation of this procedure. This research proposes the use of Extreme Learning Machines (ELM) in sorting and classifying tomatoes. The study utilizes color features as input to the ELM Network. The study proposes the use of L\*, a\*, Hue, and red-green difference. The experiment results show that ELM can be an efficient method in the classification of tomatoes. Also, the training time for ELM is fast compared to other researches. The proposed color features also performed better than other color feature combinations.

Keywords: *Tomato Classification, Image Processing, Color Space, Machine Learning, Extreme Learning Machines.*

1. Introduction

Tomato is arguably one of the economically important fruit in the world. This is because exports of tomato and its lateral products such as ketchup and tomato sauce have considerable income. In turn, tomato is the most investigated member of the fleshy fruit regarding fruit development and ripening [19].

Before transporting to the market, tomatoes are sorted to several grades. Different tomato products have distinct requirements for maturity to achieve standards. Therefore, tomato maturity is one of the important factors associated with the quality of processed tomato products.

Traditionally, the sorting of tomatoes is accomplished by human power. However, this type of scheme has many shortcomings like cost, time and inconsistency. Thus, in recent years, numerous studies have been conducted using computer techniques for assessing the quality of agricultural crops.

Tomato maturity can be related to the surface color. Thus, evaluating the maturity level by visual analysis is a feasible approach. With the fast advancement of technology, a number of image classification techniques have been developed. Some of the most popular techniques are the use of neural networks (NN) and support vector machines (SVM) [1] [5]. However, both of these popular learning techniques face some challenging issues such as intensive human intervention and slow learning speed.

Huang et al. [8] introduced the extreme learning machines (ELM) which overcomes some of the issues faced by both NN and SVM. This study will explore the feasibility of using ELM in the classification of tomatoes.

1.1 Tomato Maturity Stages

The surface color of a tomato is the most important external characteristic to assess maturity and is a major factor in the consumer’s purchase decision. Degree of ripening is usually estimated by color charts [11]. Based on the external color, the United States Department of Agriculture (USDA) established six ripening stages reflecting human ability to differentiate ripeness. The surface color of the tomato changes from green to red. The description of each stage is shown in Table 1.

Table 1: USDA Tomato ripening stages

|  |  |
| --- | --- |
| Stage | Description |
| Green | surface is completely green in color |
| Breaker | definite “break” in color of not more than 10% |
| Turning | change of color is more than 10% but not more than 30% |
| Pink | change of color is more than 30% but not more than 60% |
| Light Red | change of color is more than 60% but not more than 90% |
| Red | change of color is more than 90% |

1.2 Image

Digital Images are electronic representations of images that are stored in a computer [18]. They are composed of pixels (picture elements), each representing a color at a single point in the image.

Grayscale Images are images whose pixel values range from 0 to 255, which is equivalent to 8 bits. There are several methods to convert color images to grayscale. One of the most common methods is the luminance method as described in equation (1).

|  |  |
| --- | --- |
|  | (1) |

Binary images are images that have been quantized to two values, usually denoted by 0 and 1, but often with pixel values 0 and 255, representing black and white [2]. The conversion is typically done by using a threshold, i.e. pixels with value above the threshold are set to 1 (or 255), while the rest are set to 0. The binary conversion technique used in this research is the Otsu thresholding technique.

1.3 Otsu thresholding

The algorithm assumes that the image contains two classes of pixels, the foreground and background. Otsu’s thresholding method involves iterating through all possible threshold values and calculating a measure of spread for the pixel levels of each side of the threshold, i.e. the pixels either falls in the foreground or background [15]. The aim is to find the threshold value that minimizes the within-class variance (2) or maximize the between-class variance (3). The between-class variance is far quicker to calculate; therefore it is a much better approach to use.

|  |  |
| --- | --- |
|  | (2) |
|  | (3) |

1.4 Image Processing

Image processing is the manipulation of an image to produce either an image or a set of characteristics/parameters related to the image. According to [17], image processing is used in applications for two purposes: (1) Improving the visual appearance of images to a human observer, including their printing and transmission and (2) Preparing images for measurement of their features and structures which they reveal. Image processing has provoked a tremendous level of interest in the past years, making it a vast field of study. Thus, various techniques in image processing have been introduced, including visual enhancement, data extraction and the context of reproduction and transmission. These techniques include scaling and cropping.

Image scaling is the process of resizing a digital image [16]. It increases or decreases the size of the image. This process always involves a trade-off between efficiency, smoothness, and sharpness. Image size is commonly decreased or sub-sampled to fit a smaller display area. Enlarging or interpolating an image, on the other hand, is common for making smaller images fit a bigger screen.

Cropping an image removes some parts of the image to improve framing or to highlight objects of interest [6]. Usually, this is useful for discarding areas with less useful information. This can also help address the problem of unbalanced images.

1.5 Color Space

A color space, also known as color model, is a mathematical or numerical representation of a set of colors. The purpose of a color model is to facilitate the specification of colors in some standard generally accepted way. In essence, a color model is a specification of a 3-D coordinate system and a subspace within that system where each color is represented by a single point [3]. Past researches under tomato classification have utilized the RGB, HSI, and CIE L\*a\*b\* color models. All of the color spaces can be derived from the RGB information.

The red, green, and blue (RGB) color space is the most basic and well-known color model. In this model, each color appears as a combination of red, green and blue, which are called the primary colors. The combination of red, green and blue at full intensities makes white [3]. The RGB color space is widely used in computer graphics because color displays use red, green, and blue to create the desired color. Also, this model can simplify the architecture and design of the system. However, the RGB color space fails in dealing with “real world” images.

The HSI color model was developed to be more “intuitive” in manipulating colors and was designed to approximate the way humans perceive and interpret colors [3]. In this model, a color is represented by three components: Hue (H), Saturation (S), and Intensity (I). Hue defines the color itself ranging from 0 to 360 degrees. Saturation indicates the degree to which the Hue differs from a neutral gray running from 0 (no color saturation) to 1 (fullest saturation). Intensity indicates the illumination level. Equations (4) to (6) are used to convert RGB to HSI.

|  |  |
| --- | --- |
|  | (4) |
|  | (5) |
|  | (6) |

The CIELAB color model is considered to be perceptually uniform and is referred to as a uniform color model [3]. This color model is a uniform derivation from the standard CIE XYZ space. The L\* component represents Lightness, ranging from 0 to 100. The a\* and b\* components are the chrominance, sometimes referred to as red / blue and yellow / blue chrominance respectively. Equations (7) to (11) are used to convert RGB to CIELAB. First the RGB is converted to CIE XYZ and then CIE XYZ to CIELAB.

|  |  |
| --- | --- |
|  | (7) |
|  | (8) |
|  | (9) |
|  | (10) |
|  | (11) |

1.6 Neural Networks

A Neural Network (NN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information [13]. It is a collection of highly interconnected processing elements (neuron) working in unison to solve specific problems. With their remarkable ability to derive meaning from complicated or imprecise data, NNs can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques [13].

NNs are modeled as a collection of nodes connected via directed links. Each link has an associated weight that represents the strength of a connection between the connected nodes. The value of the output node is determined by an activation function. This function takes the summation of the input nodes and the corresponding weights. There are many activation functions in use. The most common is the logistic function, also known as Log-Sigmoid written mathematically as

|  |  |
| --- | --- |
|  | (12) |

NN is very popular because of its superior generalization capability. However, the major bottleneck in its application is the slow learning speed. The algorithms that are used to train NN consumes a lot of time and some require intensive human intervention.

1.7 Extreme Learning Machine

The Extreme Learning Machines (ELM) is a recent breakthrough in machine learning and data analysis. It tries to solve the pressing issues encountered by traditional learning techniques such as neural networks (NN) and support vector machines (SVM). These issues include intensive human intervention and slow learning speed. Developed by Huang et al. [4], ELM works for the “generalized” single-hidden layer feed-forward networks (SLFNs) but the hidden layer (feature mapping) in ELM need not be tuned. In NN, all hidden nodes in SLFNs need to be tuned. ELM however shows that these hidden nodes need not be tuned and can be randomly generated. In fact, all parameters of ELM can be analytically determined. The ELM algorithm is summarized on the next paragraph.

Given a training set , hidden node output function , and hidden node number *L*,



1. Randomly generate hidden node parameters (), *i* = 1 ,…, *L*.



1. Calculate the hidden layer output matrix **H**.
2. Calculate the output weight vector **β**: **β =**



Hidden node parameters and are randomly generated. The parameter corresponds to the input weights connecting the input layer and the hidden layer. The parameter corresponds to the bias for each hidden nodes. These parameters remain fixed after randomly generated.

The calculation of the hidden layer output matrix **H** is done using the hidden node output function **.**  denotes the activation function. The matrix **H** is written as:

|  |  |
| --- | --- |
|  | (13) |

is the Moore-Penrose generalized inverse of matrix , computed as:

|  |  |
| --- | --- |
|  | (14) |
|  | (15) |

If a positive value 1/*λ* is added to the diagonal of or in the calculation of , the resultant solution is more stable and tends to have better generalization performance. The formula for the calculation of the output weight is as described in equations (16) and (17):

|  |  |
| --- | --- |
|  | (16) |
|  | (17) |

2. Literature Review

There is a growing interest in the field of image classification. It has been applied to various fields and one of which is in fruit production. A lot of interests have been given to tomato classification. Researchers have used different methods of classification including colorimeters, NaiveBayes, Multilayer Perceptron, RBF Network, Neural Binary Tree, Random Tree, Random Forest, Instance Base K-Nearest Neighbor, K-Star (K\*), Neuro Fuzzy Inference, morphological features, model-based approach, LVQ Networks, and SVM.

Lopez et al. [11] used colorimeters obtained from the CIELAB color space to calculate color indexes. The relationship between these indexes and the visual color classification of tomatoes were compared. The researchers used the USDA color chart for the tomato ripening stages.

Asadollahi et al [1] used image processing techniques in classifying tomatoes. The dataset is composed of 90 images of tomato. The researchers classified tomatoes into three groups: high quality, medium quality, low quality. They compared classifier methods for data classification which include NaiveBayes, Multilayer Perceptron, RBF Network, Neural Binary Tree, Random Tree, Random Forest, Instance Base K-Nearest Neighbor and K-Star (K\*).

Iraji et al [10] used both fuzzy inference system (FIS) and adaptive neuro fuzzy inference system (ANFIS) for an accurate and appropriate decision on tomato classification. They concluded that the ANFIS has less error and is more accurate compared to FIS.

Wang et al [20] used vision-based judgment for tomato maturity under growth conditions. They used near-infrared images and RGB/HSI images. In their study, five maturity stages were used: breakers, turning, pink, light-red, and red. Tomato images were taken under natural illumination and growth conditions.

Van de Poel et al. [19] used a model-based approach to classify tomatoes according to their physiological maturity. The researchers used the concept of biological age which is based on two parameters: color (expressed as hue) and mass. The researchers utilized two datasets for their study which are the calibration dataset and application dataset.

Fojlaley et al. [5] analyzed the qualities of tomato using three different methods: Learning Vector Quantization (LVQ), multilayer perceptron (MLP), and support vectors machine (SVM). Their dataset included 142 tomato images acquired using a digital camera; 86 of which are training data and 56 were testing data. Prior to the feature extraction step, the images underwent a series of image processing techniques which include denoising and contrast improvement. The outputs are classified to three: good, fair, bad.

Followed its conception, Extreme Learning Machines (ELM) has attracted the attention of more and more researchers. As an emergent technology that overcomes some of the challenges faced by other computational intelligence techniques, ELM has certainly proven its worth. Providing better generalization performance at a much faster learning speed and with the least human intervention, ELM has been applied to a vast number of fields.

Minhas et al. [12] used ELM in object detection and categorization. In their study, they combined global and local object information. To attest the credibility of their study, they used a number of common publicly available image databases such as UIUC, Caltech, MIT, GRAZ and PASCAL.

Huang et al [7] used ELM for classification problems. The ELM was extended to support vector networks. The performance of ELM was tested against 11 UCI datasets and 2 Gene expression datasets. They compared the results to SVM and SVM with ELM kernel.

Huang et al. [9] also applied ELM to regression and multiclass classification problems. To test the performance of ELM, results were compared to the least square support vector machine (LS-SVM) and proximal support vector machine (PSVM). In order to verify the performance, a wide variety of datasets have been tested including 12 binary classification cases, 12 multi-classification cases, and 12 regression cases.

3. Proposed Approach

The flow of the processes for this study is illustrated in Fig. 1. The tomato image undergoes a number of preprocessing steps before it is fed into the ELM network.

Fig. 1 The pipeline of TomatoX.

3.1 Tomato Image

The input is a tomato image of any dimension. The tomato images used in this study are acquired using a digital camera. Four features will be extracted from the RGB, HSI, and CIELAB color space. These features are L\*, a\*, Hue, and red-green difference. These features represent the input vector for the ELM network.

3.2 Scaling

Since the images taken from a digital camera are quite large, they must be reduced appropriately. Tomato images will be reduced to a 200 x 200 dimension. In this way, some of the processing steps will be much faster.

3.3 Blue Channel

The next step is to extract the values from the blue channel. The blue channel is the one being manipulated in the other steps. We can easily accomplish this step since the blue channel is just a subset of the RGB color space. We just set the values of the red and green channel to zero. In turn, only the blue channel will be retained.

3.4 Grayscale Conversion

After extracting the blue channel, the image will be converted to grayscale. The luminance method shown in equation (1)will be used to convert the pixels of the blue channel image to grayscale. A grayscale image contains the same values for the red, green, and blue channel.

3.5 Masking

The next step is to make the binary mask using the Otsu thresholding technique. First, the histogram of the grayscale image is computed. It is usually represented as a one dimensional array. The indexes of the array indicate the pixel values of the image. Since the image is grayscale, the array is only of size 256.

The histogram is calculated by iterating through all the pixels of the image and incrementing the array index corresponding to the pixel value. Fig. 2 shows the code snippet of calculating the histogram.

Fig. 2 Calculating the histogram.

After the computation of the histogram, the optimal threshold can be calculated using Otsu’s method. This method will iterate from 0 to 255 finding the threshold that maximizes the between-class variance using equation (3). This threshold will determine which pixels belong to the foreground and which pixels belong to the background. Pixels to the left of the threshold are considered as foreground (1 or 255) while the pixels to the right belong to the background (0).

After creating the binary mask, the next step is the masking operation. This is done by applying the binary mask to the original tomato image, i.e. multiplying the pixels of the images. If the pixel value of the binary mask is 1, the product is the same as the original pixel. On the other hand, if the pixel value of the binary mask is 0, the product will be 0 which is the same as the binary mask pixel.

3.6 Cropping

The image needs to be cropped since the background is of no use. The background pixels will just contribute to the processing time and may affect the features to be extracted. To remove the background, the bounds of the tomato (foreground) must be calculated. This can be done easily since the background was already replaced by a black color.

3.7 Feature Extraction

Before the cropped image undergoes feature extraction, it must be resized to a 64 x 64. The features that will be extracted from the image are the average L\*, average a\*, average Hue (H), and average red-green difference (R-G). Extracting the RGB features are the easiest since the input image is already in the RGB format. However, for the HSI and CIE L\*a\*b\* features, the RGB image must first be converted to the respective color spaces. The conversion to the HSI color model is done by using equations (4)**,** (5) and (6)while the conversion to the CIE L\*a\*b\* color model is done using equations (8), (9), (10).

3.8 ELM Training

The training set for this study is consisting of 70% of the dataset. The network structure for TomatoX is shown in Fig. 3.

Fig. 3 ELM Network Structure of TomatoX.

The number of input nodes corresponds to the number of color features. In this study, the four (4) features extracted from the tomato image constitute the number of input nodes.

The number of hidden nodes is specified by the user. According to [8], the number of hidden nodes in the basic ELM algorithm should not exceed the number of samples, i.e. where corresponds to the number of hidden nodes and denotes the number of samples. In this study, the number of hidden nodes should not exceed 70% of the dataset.

The number of output nodes corresponds to the maturity stages of the tomato. In this study, six stages of tomato maturity are distinguished, namely: green, breakers, turning, pink, light red, red. Therefore, the number of output nodes is equal to six (6).

The first step in the ELM Training is to randomly generate hidden node parameters, (input weights connecting the input layer and hidden layer) and (bias of each hidden node), with values ranging from -1 to 1.

After calculating the hidden node parameters, we can now proceed to the next step which is the calculation of the hidden layer output matrix using the expression shown in equation (13)**.** The activation function used in this study is the Log-Sigmoid function calculated using equation (12)**.** Then we can proceed to the calculation of the output weight vector which is derived using and the target output vector

Unlike the traditional learning algorithms, ELM happens in just one cycle. The hidden node parameters and and the output weight vector are then concatenated and stored in a file. This file is known as a classifier which is used in the classification of a tomato.

3.9 Tomato Classification

The hidden node parameters and the output weight vector are first retrieved from the classifier specified by the user. The features extracted from the tomato will serve as the input nodes of the neural network. The hidden layer output matrix is computed in the same way as in the previous section. The values of the output nodes are calculated by multiplying and the output weight vector . The output values are normalized to determine which tomato stage corresponds to the inputted values. Normalization is done by getting the index with the maximum value.

4. Experiments and Results

The dataset is composed of six hundred (600) images, one hundred (100) images per tomato stage. The images were acquired using a digital camera and their maturity stage was manually classified. The effectiveness of the ELM classifier will be tested against this manual classification. The dataset was divided into two subsets, the train set and the test set. The train set is composed of seventy percent (70%) of the dataset while the test set is composed of the remaining thirty percent (30%). The division is done by selecting 70% of the images per tomato stage to be used as train set and the remaining 30% for the test set. Since every stage is composed of 100 images, 70 of this are randomly selected to be used in training and the remaining 30 for testing.

To assess the performance of ELM in the evaluation of tomato maturity, a standard ELM configuration was defined. Table 2 shows the standard parameter values of the ELM parameters.

Table 2: ELM Standard parameter values

|  |  |
| --- | --- |
| Parameter | Standard Value |
| Number of Hidden Nodes | 105 |
| Regularization Coefficient (C) | Not set |

Four (4) experiments were conducted in this study. The experiments are as follows:

1. Experiment I – comparison of fixed and random dataset
2. Experiment II – variation in the number of hidden nodes
3. Experiment III – variation in the values of the regularization coefficient
4. Experiment IV – variation in the number of input color features

For statistical acceptability, all experimental setups were subjected to 30 trials. In all the experiments, the ELM network training time reached a minimum of 0.2 seconds and a maximum of 4 seconds. This is due to the fact that the ELM algorithm happens in just one sweep, i.e. no iterations are needed in training.

4.1 Experiment I

For experiment I, two datasets were compared, the fixed data and the random data. For the fixed data, a fixed train and test set were used for all 30 trials. On the other hand, for the random data, a random train and test set were used for every trial. The standard ELM configuration was used in this experiment. Both the train and test set were used in the comparison. Table 3 shows the results for this experiment.

Table 3: Results for fixed and random data

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Fixed | | Random | |
|  | **Test set** | **Train set** | **Test set** | **Train set** |
| Mean | 97.593% | 98.865% | 97.185% | 98.905% |
| Median | 97.778% | 99.048% | 97.222% | 98.929% |
| Minimum | 95% | 98.333% | 93.333% | 97.619% |
| Maximum | 99.444% | 99.524% | 100% | 99.524% |
| STDEV | 1.266 | 0.399 | 1.458 | 0.445 |
| Variance | 1.604 | 0.159 | 2.127 | 0.198 |

For the test set, the fixed data has a higher average than that of the random data but the random data got 100% accuracy on one of the trials. The accuracy for the random data could go as low as 93.333336%. This is due to the fact that since a random training and test set is used for every trial, there is a possibility that the images that are more difficult to classify were on the test set while the easier ones were on the training set. In other words, the classifier was not trained well to handle more complicated inputs. There is also a possibility that the opposite situation will happen, i.e. the images that are easier to classify were on the test set while the difficult ones were on the training set. This can be seen on the results for the train set, the random data has a higher average compared to the fixed data.

Based on the results, it shows that the ELM classifier is robust since it was able to classify with high accuracy regardless how the datasets were generated. For the random data, the classifier could correctly classify all data. The classifier could even handle complex data with little impact on the accuracy as can be seen in the results for the random data.

4.2 Experiment II

For experiment II, the number of hidden nodes was varied according to the number of samples. According to the interpolation theorem [8], the maximum number of hidden nodes required is not larger than the training samples. The maximum number of hidden nodes used for this experiment is equal to the number of training samples. Since the dataset is composed of 600 images, 420 images were used as training samples. The variation in the number of hidden nodes used was N, 0.75N, 0.5N, 0.25N where N is the number of samples. Table 4 shows the variation in the number of hidden nodes.

Table 4: Varying the number of Hidden Nodes

|  |  |
| --- | --- |
| Set | Number of Hidden Nodes |
| A | 105 |
| B | 210 |
| C | 315 |
| D | 420 |

The fixed data used in Experiment I is also used in this experiment. Both the train and test set were used in testing the efficiency of the classifier. Table 5 and 6 shows the results of the test and train set respectively.

Table 5: Results of varying the number of hidden nodes for the test set

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | A | B | C | D |
| Mean | 97.593% | 96.796% | 94.667% | 87.463% |
| Median | 97.778% | 96.944% | 94.444% | 88.056% |
| Minimum | 95% | 94.444% | 92.222% | 72.222% |
| Maximum | 99.444% | 98.889% | 97.222% | 93.889% |
| STDEV | 1.266 | 1.069 | 1.12 | 4.494 |
| Variance | 1.604 | 1.143 | 1.439 | 20.193 |

Table 6: Results of varying the number of hidden nodes for the train set

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | A | B | C | D |
| Mean | 98.865% | 99.556% | 99.714% | 99.952% |
| Median | 99.048% | 99.524% | 99.762% | 100% |
| Minimum | 98.333% | 99.048% | 99.286% | 99.762% |
| Maximum | 99.524% | 99.762% | 100% | 100% |
| STDEV | 0.399 | 0.205 | 0.170 | 0.097 |
| Variance | 0.159 | 0.042 | 0.029 | 0.009 |

As shown in Table 5, for the test set, as the number of hidden nodes is increased, the accuracy of the classifier tends to decrease. This is because as the number of hidden nodes approaches the number of training samples, the ELM becomes a singular least-squares problem and gives an unstable solution. The proof of having unstable solution can be depicted in the results when the number of hidden nodes is 420. The standard deviation is at highest and the accuracy could go as low as 72.22222%. The highest accuracy for this set was attained when the number of hidden nodes is 105, correctly classifying 179 out of 180 images achieving a success rate of 99.44444%.

As shown in Table 6, for the train set, as the number of hidden nodes is increased, the accuracy of the classifier increases accordingly. This can be contrasted to the results obtained in the test set. The reason behind this is that as the number of hidden nodes approaches the number of training samples, the matrix **H** becomes square and invertible. Thus, according to [8], SLFNs can approximate these training samples with zero error. In fact, for this set, when the number of hidden nodes is 420, the maximum accuracy of the classifier is 100%.

4.3 Experiment III

The regularization coefficient (C) can be any positive number. For this experiment, the value of the regularization coefficient (C) is varied from 1 to 50. The standard number of hidden nodes was used in this experiment. The fixed dataset used in experiment I was also used in this experiment but only the test set was used in testing the classifier, i.e. for every regularization coefficient (C), the train set was used for training but only the test set was used in testing the accuracy of the classifier. This experiment was carried out to see if the introduction of the regularization coefficient will improve the performance of ELM in tomato classification. Table 7 shows the results of varying the values of the regularization coefficient (C). To avoid cluttering of data, only values of C up to 20 is shown.

Table 7: Results of varying the values of the regularization coefficient

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | No C | 1 | 10 | 20 |
| Mean | 97.593% | 98.259% | 98.13% | 98.074% |
| Median | 97.778% | 98.333% | 98.333% | 98.333% |
| Min | 95% | 93.889% | 95.556% | 94.444% |
| Max | 99.444% | 100% | 99.444% | 100% |
| STDEV | 1.266 | 1.21 | 0.905 | 1.165 |
| Variance | 1.604 | 1.463 | 0.819 | 1.357 |

As shown in Table 7, the results for all values of C have greater accuracy compared to the result obtained without the introduction of C. This suggests that the introduction of the regularization coefficient (C) further improves the classification capability of ELM. However, as shown in Table 7, an increase in the value of C does not necessarily mean an increase in the accuracy. In fact, 100% accuracy was achieved when the value of C is 1 and 20. The results also show that the introduction of a regularization coefficient makes the classifier more stable as can be seen on the values of the STDEV.

4.4 Experiment IV

For the last experiment, the number of input features was varied. Table 8 shows the variation in the combination of the input features. The fixed data used in Experiment I is also used in this experiment. The standard ELM configuration was used in this experiment.

Table 8: Varying the input features

|  |  |
| --- | --- |
| Set | Color Feature (s) |
| A | R-G |
| B | H, R-G |
| C | L\*, a\* |
| D | L\*, a\*, H, R-G |
| E | R, G, R-G, H, a |

As shown in Table 8, each set has a different network structure since the input features correspond to the number of input nodes. The result of varying the input features is shown in Table 9.

Table 9: Results of varying the input features

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E |
| Mean | 96% | 94.4% | 96.7% | 97.6% | 95.8% |
| Median | 96.1% | 94.7% | 96.7% | 97.8% | 96.1% |
| Min | 94.4% | 92.2% | 95% | 95% | 91.1% |
| Max | 96. 7% | 96. 7% | 98.3% | 99.4% | 98. 9% |
| STDEV | 0.68 | 1.384 | 0.711 | 1.266 | 2.02 |
| Variance | 0.463 | 1.916 | 0.505 | 1.604 | 4.082 |

As shown in Table 9, the results show that set D, containing the proposed input features performed best amongst all the setups. In fact, it can correctly classify 179 out of 180 images achieving an accuracy of 99.44444%. The results also show that the use of more input features does not necessarily mean an increase in the accuracy. In fact, set A containing only one input feature performed better than set B that contains two input features and set E that contains five input features.

For comparison, the result of this study is compared to the work of Opeña [14]. The results are not compared to other researches since each research used a different dataset.

Table 10: Comparison to TotoBee[14]

|  |  |  |
| --- | --- | --- |
|  | Current Research | TotoBee[14] |
| Accuracy | 97.593% | 98.19 % |
| Training Time (sec) | 1.89 | 153.46 |

As shown in Table 10, the accuracy of Opeña [14] is larger than that of this research but only with a small difference. On the other hand, the training time of this research is smaller than that of Opeña [14]. Time is a necessity in industrial applications. The results of this research offer faster training time with a small decrease in accuracy.

5. Conclusion and Future Works

Extreme Learning Machines (ELM) as an emergent technology has attracted a lot of attention from researchers. It has been applied to various problems and it has shown great results. In this study, ELM was implemented to solve the tomato classification problem. Experimental results show that ELM can be an efficient method in the classification of tomatoes.

In the first experiment, it has been shown that the ELM classifier is robust. It can handle complex data with little impact on the accuracy. In fact, for the random data, it correctly classified all tomato images.

In the second experiment, it has been shown that the number of hidden nodes can affect the performance of ELM. As the number of hidden nodes is increased, the accuracy for the train set increases while the accuracy for the test set decreases.

In the third experiment, the introduction of a regularization coefficient (C) further improves the generalization capability of the ELM. However, based on the results, an increase in the value of C does not necessarily mean an increase in the accuracy. It has also been shown that the introduction of a regularization coefficient makes the classifier more stable.

In the last experiment, it has been shown that the input features have a significant effect on the performance of the ELM. However, an increase in the number of input features does not guarantee a good result. Set A containing only one input feature has better performance than set B and set C containing two and five input features respectively. Overall, set D containing the proposed input features performed best amongst all the setups. It correctly classified 179 out of 180 images achieving an accuracy of 99.44444%.

Future works on this study could include the extension of this study to real-time tomato classification, i.e. the use of machines to pick tomatoes and directly classify their maturity stage. One could also make use of the Online-Sequential ELM (OS-ELM) which eliminates the need to retrain the whole dataset if the dataset is updated. Interested researchers could even apply the ELM to other vegetables or fruits especially those grown widely in the Philippines such as bananas.

References

[1] Asadollahi, H., Kamarposhty, M., Teymoori, M.: Classification and Evaluation of Tomato Images Using Several Classifier. In Proceedings of the 2009 International Association of Computer Science and Information Technology – Spring Conference (IACSIT-SC '09). (2009) 471 – 474

[2] Binary Images. From, http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL\_COPIES/OWENS/LECT2/node3.html (Accessed: March 2013)

[3] Color Models. From, http://software.intel.com/sites/products/documentation/hpc/ipp/ippi/ippi\_ch6/ch6\_color\_models.html (Accessed: March 2013)

[4] Extreme Learning Machines. From, http://www.ntu.edu.sg/home/egbhuang/ (Accessed: March 2013)

[5] Fojlaley M., Moghadam, P., Nia, S.: Tomato Classification and Sorting with machine vision using SVM, MLP, and LVQ. International Journal of Agriculture and Crop Sciences. (2012) 1083 – 1088

[6] How to Crop Images, From, http://digital-photography school.com/cropping-for-impact (Accessed: March 2013)

[7] Huang, G.B., Ding, X., Zhou, H.: Optimization method based extreme learning machine for classification. Neurocomputing. (2010) 155 – 163

[8] Huang, G.B., Wang, D.H., and Lan, Y.: Extreme Learning Machines: A Survey. International Journal Machine Learning & Cybernetics. (2011) 2:107 – 122

[9] Huang, G.B., Zhou, H., Ding, X., Zhang, R.: Extreme Learning Machine for Regression and Multiclass Classification. IEEE Transactions on Systems, Man, and Cybernetics, Part B 42 (2). (2012) 513 – 529

[10] Iraji, M., Tosinia, A.: Classification Tomatoes on Machine Vision with Fuzzy the Mamdani Inference, Adaptive Neuro Fuzzy Inference System Based (Anfis-Sugeno), vol. 5 (11). (2011) 846 – 853

[11] Lopez Camelo, A., Gomez, P.: Comparison of color indexes for tomato ripening. Horticultura Brasileira, vol. 22, n.3. (2004) 534 – 537

[12] Minhas, R., Mohammed, A., Wu, J.: A fast recognition framework based on extreme learning machine using hybrid object information. Neurocomputing. (2010) 1831 – 1839

[13] Neural Networks. From, http://www.doc.ic.ac.uk/~nd/surprise\_96/journal/vol4/cs11/report.html (Accessed: March 2013)

[14] Opeña, H. J.: ToToBee: Tomato Maturity using Artificial Bee Colony Algorithm and Artificial Networks.

[15] Otsu Thresholding. From, http://www.labbookpages.co.uk/software/imgProc/otsuThreshold.html (Accessed: March 2013)

[16] Quick image scaling algorithms. From, http://www.compuphase.com/graphic/scale.htm (Accessed: March 2013)

[17] Russ, J.: The Image Processing Handbook, Sixth Edition. CRC Press (2011)

[18] Understanding digital images. From, http://www.reading.ac.uk/internal/using-images/UnderstandingDigitalImages/img-understandhome.aspx (Accessed: March 2013)

[19] Van de Poel, B., Bulens, I., Hertog, M., Van Gastel, L., De Proft, M., Nicolai, B., Geeraerd, A.: Model-based classification of tomato fruit development and ripening related to physiological maturity. Postharvest Biology and Technology, vol. 67. (2012) 59 – 67

[20] Wang, X., Mao, H., Han, X., Yin, J.: Vision-based judgment of tomato maturity under growth conditions. African Journal of Biotechnology, vol. 10 (18). (2011) 3613 – 3623