From pixels to patterns: AI-driven image analysis in multiple domains

Javanmardi, S.

CHAPTER 2

Corn Seed Feature Extraction via Deep Learning

This chapter is based on the following publication:

S. Javanmardi, S.-H. M. Ashtiani, F. J. Verbeek, and A. Martynenko, Computer-vision classification of corn seed varieties using deep convolutional neural network, Journal of Stored Products Research, vol. 92, p. 101800, 2021.

2.1 Chapter Summary

Building upon the fundamental concepts and research inquiries established in the preceding chapter, we now move to the precise classification of corn seed varieties. This chapter delves into the utilization of Deep Learning to analyze single-instance images of corn seeds, extracting and identifying those features that critically influence classification accuracy. This line of inquiry is driven by the vital necessity for seed producers to guarantee varietal integrity and maximize agricultural yield. Conventional techniques, reliant on basic computer vision and elementary feature extraction, have fallen short in achieving the high accuracy required for such classification tasks. In this context, we introduce an application of Convolutional Neural Network (CNN) as advanced feature extractors. The CNN adeptly delineate rich feature sets that are subsequently classified through a suite of advanced algorithms. Our extensive experimental analysis reveals that leveraging features extracted by CNN and using Artificial Neural Network (ANN) as classifier noticeably enhances classification performance over the other traditional approaches for corn seed variety classification.

2.2 Introduction

Classification of seed varieties is of paramount importance for seed producers and farmers to maintain the purity of a variety and crop yield (Shouche et al., 2001), (Taner et al., 2018). Moreover, premium seed varieties are more expensive due to their potential to increase production and profits. This issue is crucial for corn being one of the most commercial crops worldwide (Wakholi et al., 2018). However, corn seeds of different varieties are very similar, with significant overlap in both morphology and color features. These factors have created an apt opportunity for some opportunistic seed mills to illegally market low-quality seed varieties as high-quality and raise their profit margin Jia et al. (2015). This reckless behavior can compromise investors' interests and disrupt the seed market. Therefore, the accurate classification technique is vital to develop the seed market and protect the interests of farmers and owners of premium quality seed varieties.

Traditionally, a seed variety is identified through visual inspection Vithu and Moses (2016), Ansari et al. (2021). High error rates, low precision, and time-consuming nature are some of challenges of this approach, especially for identical species (Pourreza et al., 2012), (Taner et al., 2018). Moreover, regular access to an agriculture expert is not possible when consultation is required (Kurtulmuş and Ünal, 2015), (Khan et al., 2019). High-performance liquid chromatography, gas chromatography-mass spectrometer (Qiu et al., 2018), seed protein electrophoresis (Rogl and Javornik, 1996), and DNA molecular markers (Hoffman et al., 2003) are some of the standard analytical methods used to classify plant varieties. In most cases, despite high accuracy, these methods are destructive, hazardous to human health, time-consuming, complicated, and expensive, with little chance of reproducibility (Ma et al., 2013), (Bakhshipour et al., 2018).

In today's world, employing an automated system for non-destructive and accurate seed classification is critical. In this regard, many studies present techniques for non-destructive seed classification, such as magnetic resonance imaging, electronic tongue, acoustic, electronic nose, and computer vision (Xia et al., 2019). Among these methods, computer vision and image processing can classify crops at a low cost with high analytical and computational power (Patrício and Rieder, 2018), (Ropelewska, 2020). Computer vision-based classification consists of four blocks: image preprocessing, segmentation, feature extraction, and classification (Sharif et al., 2018), where feature extraction has a significant effect on classification accuracy (Iqbal et al., 2018). As a result, several feature extraction and machine learning techniques have been proposed for seed variety classification. The extracted features include color features (Li et al., 2019), texture features (Zhao et al., 2011), shape-based features (Ma et al., 2013) , and their combination (Kiratiratanapruk and Sinthupinyo, 2011). Different classifiers based on k-means clustering (Ma et al., 2013), backpropagation neural network (BPNN) (Li et al., 2016), Genetic Algorithm-Support Vector Machine (GA-SVM) (Zhao et al., 2011), maximum likelihood classifier (Li et al., 2019) and so on, have been probed. Previous research has shown that the computer vision technique works well for the classification of seeds with different visual appearances, such as different species, damaged or defective seeds, or foreign premises.

2.2.1 Problem statement

The classification of corn seed varieties with similar visual appearance is still a tremendous challenge. Widespread use of hybridization has recently led to the development of multiple corn seed varieties. It raised a problem of their classification, which has become an increasingly challenging task due to their extreme similarities, especially for the same mass-tone attune seeds. Previous research focused on extracting the hand-crafted features from a specific section (germ side and/or endosperm side) of corn seeds (Yang et al., 2015), (Xia et al., 2019), (Li et al., 2019) to address this problem. Unfortunately, this classification technique requires a particular seed orientation in a sorting system, making it difficult for industrial scaling.

2.2.2 Motivation and Contributions

Our research was motivated by the hypothesis that corn seed classifying could be dramatically improved using higher-level features with more discriminative information. In recent years, a tendency to substitute the classical feature extraction methods with deep learning methods, especially convolutional neural networks, has emerged. CNNs are replacing traditional methods that extracted low-level features from images, by automatically and hierarchically extracting robust high-level features (Zhang et al., 2019).

Numerous studies have demonstrated that the use of CNNs as a generic extractor can significantly improve the accuracy of computer vision tasks compared to traditional feature-engineered methods Kozłowski et al. (2019). For example, (Zhou et al., 2017) optimized the VGG Net model structure to extract features from the images of main tomato parts such as fruit, flower, leaf, and stem, using an 8-layers CNN. (Khan et al., 2019) introduced a new approach for the classification of plants' diseases based on the four-step algorithm. In the first step, disease regions are segmented with the help of the correlation coefficient method. After that, two pre-trained models (VGG-16, Caffe AlexNet) are used for the extraction of deep features. The GA is developed to select the most discriminant features. Finally, the selected features are classified by multi-class SVM and achieved 98.6% accuracy.

In another study, (Quan et al., 2019) adopted a Faster R-CNN model to automatically extract image features and detect maize seedlings during different growth stages. In the research of (Özkan et al., 2019), a CNN architecture VGG16 was used to extract features, which have been used as inputs for an SVM classifier to identify 40 different wheat grain varieties. They claimed 100% accuracy of the proposed model. (Liao et al., 2019) investigated the feasibility of haploid corn seeds classification, using hyperspectral images. A VGG-19 network was used to extract the image properties. They achieved a correct classification rate of 96.32%. (Zhu et al., 2019) extracted the characteristics of seven cotton seed varieties using self-designed CNN and ResNet models. They utilized partial least squares discriminant analysis (PLS-DA), logistic regression (LR), and SVM models to classify the seeds and reported an accuracy of 80%. (Khan et al., 2019) presented a new method that used fine-tuned VGG-s and AlexNet to extract the deep features and multi-SVM to detect the fruit diseases with an accuracy greater than 97%. (khan et al., 2020) introduced an automated approach for cucumber leaf disease classification. The authors employed two pre-trained models, i.e., VGG-19 and VGG-M for feature extraction and multi-class SVM for classification, which achieved a maximum classification accuracy of 98.08%.

In this study for the first time, CNN was used as a feature extractor for the classification of nine corn seed varieties regardless of their orientation on the conveyor. The major contributions of this research are:

- Both low-level visual (color, morphology, texture) features and high-level (CNN-extracted) features of corn seeds have been used for classification.
- The effect of input features (hand-crafted, CNN-extracted, or/and their combination) on the accuracy of classification has been studied.
- The efficiency of different machine learning classifiers, such as an Artificial Neural Network, cubic SVM, quadratic SVM, weighted (k-Nearest-Neighbor) kNN, boosted tree, bagged tree, and Linear Discriminant Analysis (LDA) for corn seed classification has been evaluated.

This Chapter is organized as follows: Section 2.3 introduces the research methodology, including a detailed description of feature extraction techniques and classifiers. In Section 2.4, the results of feature extraction methods and classification of corn seed varieties are presented. The classification performance of proposed frameworks with different inputs and classifiers is evaluated. Finally, Section 2.5 presents a summary of current research and provides direction for future work.

2.3 Data and methods

2.3.1 Sample preparation

Nine different varieties of corn, KSC 201, KSC 704, KSC 290, KSC 380, KSC 301, KSC 400, KSC 260, KSC 647, and KSC 410, were used in this study. These varieties can be divided into three groups: 1) early maturity types that are suitable for a second planting in temperate and cold regions (KSC 201, KSC 260, KSC 410, KSC 290, KSC 380, and KSC 301), mid-maturity types that are suitable for planting in warm regions (KSC 647 and KSC 400), and late maturity type that is suitable for first planting in temperate regions and second planting in warm regions (KSC 704). In addition to maturity, the varieties differ in terms of agronomic characteristics including plant height, plant density, row spacing, germination percentage, 1000 seed weight, seed yield, and protein percentage. A detailed discussion of agronomic features is beyond the scope of this Chapter; however, more information can be found on the website www.spii.ir/en-US/DouranPortal/1/page/Home.

The corn varieties were provided by the Department of Maize and Forage Crop Research, Seed and Plant Improvement Institute, Karaj, Iran. The seeds were manually and meticulously cleaned from all foreign materials such as dust, dirt, pebbles, and chaff, as well as immature and damaged seeds. The initial moisture content was measured by drying $15\,g$ of grounded seeds at 130 °C for $1\,h$ h in an oven (Jiao et al., 2016). The measurement was repeated three times. The initial moisture content of corn seeds was about 11% (wb). In the healthy seeds set, 1000 samples from each variety were randomly selected for imaging and stored in sealed plastic packages at room temperature ($20 \pm 1\,°$C). The reason for considering this number of samples was that deep learning networks need a large number of samples for appropriate training (Wen, 2020).

2.3.2 Image acquisition and preprocessing

To provide the illumination required for imaging, a 40 W fluorescent lamp about 40 cm in diameter was mounted on the table, and color (RGB) images were captured by a digital camera (Nikon D3200). The camera lens was placed at a 25 cm distance from samples, and the focal length of the camera lens was adjusted to 7.8 mm. The white balance of the camera was calibrated before capturing images. Samples were placed on a non-reflective blue paper. For color calibration, a color chart (ColorChecker Classic, X-Rite, USA) was used. To extract the features, image segmentation, and mathematical morphology were recruited. The detection of foreground and background in an image is of paramount importance, as the background generally affects the performance of the image features analyzer (Nasirahmadi and Miraei Ashtiani, 2017). In this regard, the RGB color space was

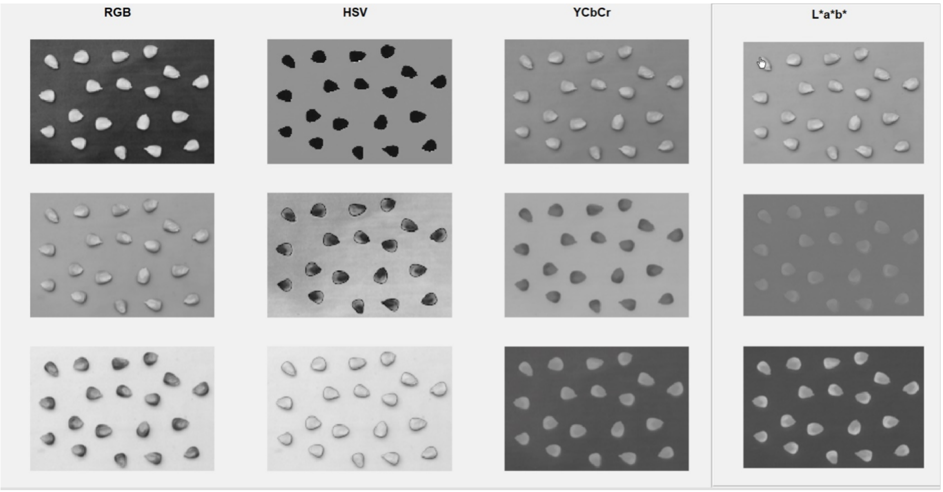


Figure 2.1: Corn seeds in different color spaces.

transformed into other color spaces such as CIE Lab, HSV, and YCbCr (shown in Figure 2.1).

After comparing four different color spaces on a set of images, the strongest contrast between the seed and the background in the Lab color space, which contained a luminance channel (L) and two chrominance channels (a and b), was obtained to be used in the next processing stages. The multi-threshold methods were used to remove the background, as described by Beyaz et al. in (Beyaz et al., 2019). The morphological filtering process was employed to eliminate any existing noise. Figure2.2 shows the whole process for image segmentation.

2.3.3 Hand-crafted features extraction

For these features, we extracted 87 features, including 6 color features, 17 morphological features, 5 Gray Level Co-occurrence Matrix (GLCM) features, and 59 Local Binary Patterns (LBP) features from images. The mean and standard deviation of three color channels in the CIELab color space resulted in six extracted color features. Binary images were used to obtain the morphological features. The seventeen measured morphological parameters are listed in Table 2.1. Moreover, two groups of texture features, including GLCM and LBP, were calculated. The details of these features are described in the following paragraphs.

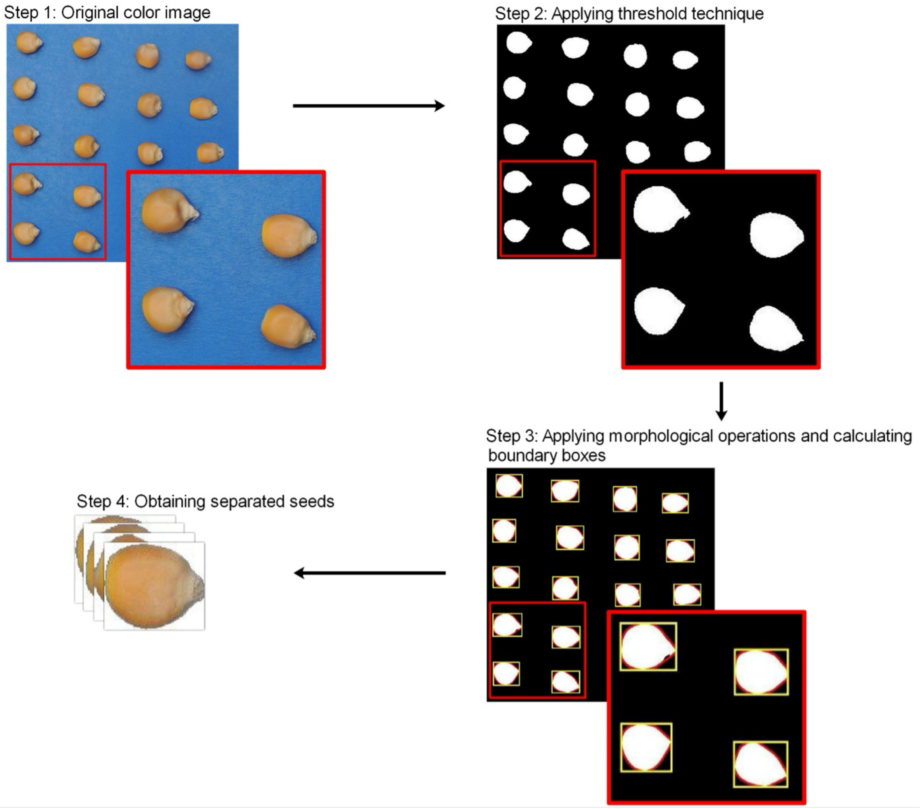

Step 1: Original color image
Step 2: Applying threshold technique
Step 3: Applying morphological operations and calculating boundary boxes
Step 4: Obtaining separated seeds

Figure 2.2: Results according to the image preprocessing and segmentation processes.

2.3.3.1 Gray Level Co-Occurrence Matrix

The GLCM is based on the estimation of the second-order statistics of the spatial arrangement of gray-level values. This matrix represents the association between two neighboring pixels where the two associated pixels have a certain gray intensity and are separated by a predefined distance and angle. The features obtained by this method are a symmetric matrix, and the matrix elements are calculated by the following formula (Kurtulmuş and Ünal, 2015):

$$p(i,j) = \frac{P(i,j)}{\sum_{i=1}^{G}\sum_{j=1}^{G} P(i,j)} \tag{2.1}$$

where G is the total number of grey levels; i and j are the pixels to be examined; $p(i,j)$ represents the co-occurrence probability between grey levels of i and j,

Table 2.1: Extracted morphological features of corn seeds.

Feature	Definition
Area	The total number of pixels inside the object
Perimeter	The number of object boundary pixels
Maximum radius	The length of the major axis of the object
Minimum radius	The length of the minor axis of the object
Solidity	The proportion of area to a convex hull
Extent	The ratio of pixels in the region to pixels in the bounding box
Eccentricity	The ratio of the distance between the foci of the ellipse and its major axis length
Equivalent diameter	The diameter of the circle whose area is equal to the area of the object
Sh_1	$2\sqrt{A/\pi}$
Sh_2	$P/2\sqrt{\pi A}$
Sh_3	$4\pi A/P^2$
Sh_4	M/A
Sh_5	A/M^3
Sh_6	$4A/\pi M^2$
Sh_7	$4A/\pi mM$
Sh_8	P^2/A
Sh_9	$P - P\sqrt{P^2 - 4\pi A}/P + P\sqrt{P^2 - 4\pi A}$

Sh: shape factor, A: area, P: perimeter, M: major diameter, m: minor diameter

$P(i,j)$ is the number of grey-level co-occurrence matrices. Five texture features from the GLCM of each image were extracted, as shown in Table 2.2. In these equations, μ_i, μ_j, σ_i and σ_j are the means and standard deviation of p_i and p_j.

2.3.3.2 Local Binary Patterns

The LBP descriptors reflect the local texture features of an intensity image by comparing each pixel with its neighborhood pixels. For each pixel in an image, the central pixel value is compared with its eight neighborhood pixels. If the neighborhood pixel is greater than or equal to the central pixel value, a binary value of 1 is generated for the neighborhood pixel; otherwise, a value of zero is considered. By concatenating these binary values in a clockwise direction, a binary

Table 2.2: Texture features extracted from GLCM matrix.

Feature	Equation
Contrast	$\sum_{i=1}^{G}\sum_{j=1}^{G} \lvert i-j\rvert^2 p(i,j)$
Correlation	$\sum_{i=1}^{G}\sum_{j=1}^{G} \frac{(i-\mu_i)(j-\mu_j)p(i,j)}{\sigma_i\sigma_j}$
Energy	$\sum_{i=1}^{G}\sum_{j=1}^{G} p(i,j)^2$
Homogeneity	$\sum_{i=1}^{G}\sum_{j=1}^{G} \frac{p(i,j)}{1+\lvert i-j\rvert}$
Entropy	$-\sum_{i}\sum_{j} p(i,j)\log(p(i,j))$

code is obtained. Then, to calculate the LBP value of the central point, this binary code is converted to a decimal value (Hu et al., 2020). The whole procedure of LBP can be defined as follows (Kurtulmuş and Ünal, 2015), (Adeel et al., 2019):

$$LBP_{M-N} = \sum_{n=0}^{M-1} s(g_n - g_k)2^n, \quad s(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases} \tag{2.2}$$

where M is the number of pixels in the neighborhood, N denotes the radius, g_n is the grayscale value of the neighborhood pixel, g_k is the grayscale value of the central pixel, and $s(x)$ is the non-linear quantization function.

2.3.4 The convolutional neural network features extraction

The structure of a CNN model consists of three main neural layers: convolution, pooling and fully connected, each of them has different tasks in the network architecture. The kernel of the CNN structure is the convolutional layer that performs the heaviest computational operations. The convolution layers closer to the input extract basic features such as edges of different orientations, while deeper convolution layers extract complex and abstract features such as specific subsampled object regions (Kozłowski et al., 2019). To capture more complex features of the input image and increase the nonlinearity of the deep learning structure, convolutional layers are often followed by activation layers. To reduce

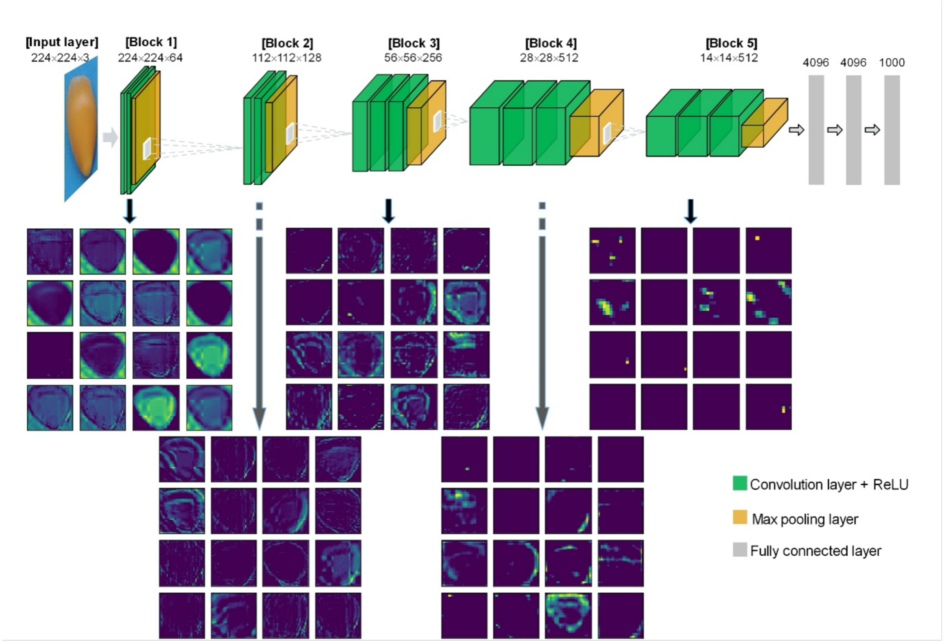


Figure 2.3: The overall structure of the VGG-16 for feature extraction. The figure also shows the visualization of the features of different blocks.

the number of parameters and computational complexity of the model, a pooling layer is placed between successive convolutional layers. As a result, this layer helps to prevent overfitting and improve generalization (Zhu et al., 2019). Fully connected layers are always placed at the end of the network that processes features imported from previous layers. The task of this layer is to convert the feature map into one-dimensional feature vectors (Anubha Pearline et al., 2019).

The DCNN architecture established for the research presented in this article is shown in Figure 2.3 The transfer learning method was employed to extract features using the VGG-16 model pre-trained on the ImageNet data set. The pretrained model weights were used as initial weights in deep learning architecture to extract features from the input image. VGG-16 is one of two VGG architectures introduced by (Simonyan and Zisserman, 2014), which consists of 13 convolutional layers, 5 pooling layers, and 3 fully connected layers. In each convolution layer, a 3×3 multiple filters with a stride of 1 pixel is used. Many nonlinear functions can be used in pooling layers such as max-pooling, average pooling, and L2-norm pooling.

In this study, max-pooling was adopted due to its simplicity and ability to maintain representative features (Huang et al., 2019). All max-pooling layers were set to a 2 × 2 window size and a stride of 2. There are several activation functions commonly used in CNNs, such as the Rectified Linear Unit (ReLU), sigmoid, and hyperbolic tangent functions. In each convolution layer, ReLU was used as an activation function. Since the ReLU can mitigate the gradient disappearance problem and provide more optimal error transmission compared to the sigmoid function (Chen et al., 2019). This function performs the following mathematical operations on each input data:

$$f(x) = \begin{cases} x, & \text{if } x > 0 \\ 0, & \text{if } x \leq 0 \end{cases} \tag{2.3}$$

where x shows the feature values of the neurons. The network input layer size was a 224 × 224 × 3 matrix, where 3 represents three RGB image channels. The convolutional filter numbers of $\text{Conv1}, \text{Conv2}, \ldots, \text{Conv5}$ were 64, 128, 256, 512, and 512, respectively. After each module, the feature map size was reduced by half so that the feature map size was 224 × 224 in the first layer and 7 × 7 in the last layer (Figure 2.3). The successive reduction in size and increased number of feature maps in the upper layers provided a wide variety of more specific and complex features. At the end of the network, there were three fully connected layers, where the first two layers consisted of 4096 neurons, and the third was a 1000 fully-connected softmax layer. This resulted in a total of 4096 CNN features from each image. The main parameters of the proposed CNN model are shown in Table 2.3.

Table 2.3: Specific parameters of the CNN architecture.

Factor	**Value**
Image input size	224 × 224 × 3
Depth	16
Optimizer	RMSprop
Loss function	cross-entropy
Max epochs	100
Batch size	32
Learning rate	0.01
Parameters	138 M

To investigate the feasibility of discrimination improvement, CNN-extracted features were also combined with hand-crafted features and fed to the classifiers.

2.3.5 Machine learning classifiers

In the classification phase of corn varieties, the extracted features were used to train the classifiers. To find the most suitable classification algorithm, the performance of seven classifiers, as described in the following sections, has been compared. To develop classification models, the data sets were randomly categorized into training (75%), and testing (25%) subsets. Within the training set, 10-fold cross-validation was applied to optimize parameters and estimate the prediction performance of the model.

2.3.5.1 Support vector machine

The SVM is a maximum margin classifier. Instead of modeling the probability distribution of training vectors, SVM attempts to separate them by directly searching appropriate boundaries between different classes. A good separation or lower generalization error of SVM is obtained by a decision line with the maximum distance from the nearest training points of each class. Using the mapping (kernel) function, this method can address complex classification problems, which are not linear in the primary dimension but could be linearized in high dimensional spaces (Jia et al., 2015). The general form of the SVM decision function is as follows (Huang et al., 2018), (Adem et al., 2019):

$$f(x) = \sum_{i=1}^{n} y_i \alpha_i K(x_i, x) + b \tag{2.4}$$

where $x_i \in \mathbb{R}^d$ is the training vector, $y_i \in \{-1, 1\}^n$ is the label of each training case, α_i is the Lagrange multiplier, n is the number of training data, K is the kernel function, and b is the bias term.

The $f(x)$ classification function represents the distance between input data and the decision-making hyper-plane. As a rule, the sample farther from the hyper-plane is more likely to be classified correctly. Among the different types of kernel functions (linear, quadratic, cubic, sigmoid and Gaussian radial basis (RBF)), two polynomial kernels, i.e., cubic and quadratic, were used, as defined by Eq. (2.5):

$$K(x_i, x) = \left[\left(x_i^T x_j\right) + 1\right]^d \tag{2.5}$$

where d = 2 for quadratic kernel and d = 3 for cubic kernel. In this equation, x_i^T is the transpose of x_i. The parameters used in both SVM classifiers were set as follows: box constraint level: 1, kernel scale mode: automatic, and standardize data: true.

2.3.5.2 Weighted k-nearest-neighbors

The KNN classifier is a non-parametric instance-based learning algorithm. The main idea behind this algorithm is to find the closest training samples in a feature space. However, the overlapping of different classes greatly diminishes classification performance. Another limitation of this method is that it places the same weight on the class labels of each k-nearest neighbor to the observation being classified (Li et al., 2020).

To overcome these drawbacks, (Hechenbichler and Schliep, 2004) proposed a weighted KNN classifier. The weighted KNN is based on the idea that greater weights are assigned to neighbors closer to the new observation compared to those further away from the new observation (Li et al., 2020). Assigning weights to neighbors according to their distance improves classification performance compared to standard KNN. Given that training samples closer to the objects are more identical, they are more likely to be classified in the same class. In the weighted KNN method, the distances are first standardized, and then the kernel function is used to transfer distances to the weights. Details of this method can be found in (Hechenbichler and Schliep, 2004). There are several types of distance metrics and weights that can be chosen. In this study, the number of nearest neighbors was set to 10, and the squared inverse of the distance and Euclidean distance were used as the weight and the distance metric, respectively.

2.3.5.3 Artificial neural network

ANNs can intelligently extract relationships between input and output data sets, even when their associations are unknown. Among various types of neural network models, the backpropagation neural network (BPNN) is widely used as a supervised classifier due to its ease of implementation and convergence as well as proper function approximation. A BPNN model usually consists of 3 layers where input and output neurons represent predictive and dependent variables, respectively, while hidden-layer neurons are tasked with information processing and transfer to the related neurons in the network. This model uses the gradient steepest descent method to adjust neuron weights and minimize output error (Kuo et al., 2020). For example, (Chimeno-Trinchet et al., 2020) used a scaled conjugate gradient backpropagation algorithm to train the network. The optimal number of hidden layers was 6. According to the type of input data set, the optimal number of hidden layer neurons varied from 10 to 45. The number of epochs was set as 1000, and the learning rate was 0.1. Mean squared error and hyperbolic tangent (tanh) were set as a cost function and activation function, respectively. A stochastic gradient descent algorithm was utilized to accelerate the network training process.

2.3.5.4 Boosted and bagged trees

The boosted tree technique is based on the integration of a collection of weak learners such as decision trees (Moon et al., 2018). Unlike linear models, this method is capable of modeling nonlinear interactions between features and target values. In the boosted tree model, each sub-tree is sequentially constructed from the prediction residuals of the previous one. In the first step, the data is divided into two data sets at each split node, the best data segmentation is determined and deviations of experimental values from their respective means are calculated at each step of the boosting process. Given the previous sequence of trees, the next tree is fitted with new residues to find another partition that will further reduce the model error (Gupta et al., 2017).

In ensemble-based learning methods, the number and depth of trees are important parameters. Increasing the depth of trees improves the model performance, but it can also lead to the overfitting problem. In the same vein, increasing the number of trees improves the accuracy of the model but prolongs the processing time (Saeed et al., 2019). Therefore, to achieve the desired result, a trade-off between the number of trees and the maximum depth is necessary. In this Chapter, the boosted tree classifier was trained using 30 trees and a maximum tree depth of 20. The adaptive boosting (AdaBoost) ensemble algorithm was employed to enhance improvement and improve tree accuracy. The learning rate was set to 0.1 with a subspace dimension 1, and the decision tree was chosen as the learner type.

Another ensemble method used in this study was the bagged tree. This model combines several decision tree classifiers to improve prediction compared to a single decision tree classifier. Generally, it trains a set of simple classifications by replicating the training data and, following the combination of their outputs, defines the final class by voting Vivar et al. (2019). The bagged tree was using the learner type of decision tree and bag ensemble technique. Similar to the boosted tree configuration, the learning rate, the subspace dimension, the number of trees, and the maximum tree depth were set to 0.1, 1, 30, and 20, respectively.

2.3.5.5 Linear discriminant analysis

LDA, as a supervised and parametric method that utilizes a Gaussian mixture model for data generation, is another classification method employed in this study. LDA reduces data size by applying linear boundaries between groups, maximizes the distance between classes and, at the same time, minimizes variance within each class (Fogarty et al., 2020). In other words, this method uses the normal distribution-based pooled covariance matrix to map features for further segmentation of different classes (Wakholi et al., 2018).

2.3.6 Performance evaluation

In this study, accuracy is considered as the most important indicator to evaluate the performance of classification models, because the data sets were balanced (Yilmaz et al., 2020). However, in addition to the accuracy, precision, recall, and F1-score were also introduced. They are computed as follows (Beyaz et al., 2019):

$$\text{Accuracy} = \frac{n_p + n_n}{n_p + n_n + n_{mp} + n_{mn}} \tag{2.6}$$

$$\text{Precision} = \frac{n_p}{n_p + n_{mp}} \tag{2.7}$$

$$\text{Recall} = \frac{n_p}{n_p + n_{mn}} \tag{2.8}$$

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.9}$$

where:

- n_p is the number of true positives: instances correctly identified as positive.
- n_n is the number of true negatives: instances correctly identified as negative.
- n_{mp} is the number of false positives: instances incorrectly identified as positive.
- n_{mn} is the number of false negatives: instances incorrectly identified as negative.

2.3.7 Software

MATLAB R2016a (The MathWorks Inc., Natick, MA, USA) was used to perform the most image analysis tasks and build the classification models. Furthermore, we used Python 3.6 to build the CNN model. All analyses were executed on an Asus computer equipped with an Intel Core i7 processor, 8 GB of DDR4 RAM, and NVidia GeForce GTX 1050 Ti with 4 GB GPU with Windows 10 and Linux Ubuntu 16.04.

2.4 Results and discussion

After extracting features from the images of the samples, ANN, quadratic, and cubic SVMs, boosted and bagged trees, LDA, and weighted KNN classifiers were trained. The results presented in Table 2.4 show the average accuracy of classification of the algorithms on the test data.

According to Table 2.4, ANN had the best performance with color features, providing overall accuracy of 78.9%; conversely, while LDA had the worst performance with 33.9% accuracy. With LBP features the highest overall classification accuracy of 85.7% was also achieved by ANN, while the lowest classification accuracy of 49.5% was achieved by LDA. Based on hand-crafted features, ANN and cubic SVM showed almost similar classification performance, with SVM classified slightly better on the set of GLCM and morphological features. LDA had consistently the worst performance, which suggests that the linear discriminant analysis is not suitable for the classification of the corn varieties. Our findings correspond to the results of (Nie et al., 2019) and (Wakholi et al., 2018), who reported that in the classification of seeds, non-linear models like SVM had a superior performance than linear models. Although LDA was not suitable for the classification of corn seeds, this method was successful to classify other agricultural products.

In a study by (Pourreza et al., 2012) wheat varieties were classified using LDA with an average accuracy of 98.15%. The poor results in the classification of corn varieties can be explained by significant similarity in color and texture features among varieties under study. The comparison of all tested classifiers showed that among hand-crafted features, morphological and LBP features had the least and most discriminatory powers, respectively, in the classification of corn varieties. The average accuracy of classification based on morphological features was lower than 45% because of the high similarity between morphological features of corn varieties. In contrast to our results, (Liu et al., 2005) observed that rice seed varieties could be classified based on morphological features with a satisfactory accuracy of 84.83%. This can be because of the more regular morphological form of rice seeds compared with corn seeds Yang et al. (2015).

By comparing the results presented in Table 2.4, we can see that fusion of all low-level features did not improve the accuracy of classification. However, introducing CNN significantly improved the accuracy of classification. For example, the highest classification accuracy 98.1% was achieved by ANN using the features extracted by CNN, whereas the lowest classification accuracy was 81.7% by the boosted tree. This study also showed the effect of combining CNN-extracted features with hand-crafted features. Generally, adding CNN-extracted to hand-crafted features resulted in better performance of models compared to using only hand-crafted features. For example, the fusion of color and CNN features increased the average accuracy 1.7 times in comparison with the models that used only color features as inputs. A similar effect was observed in the cases CNN + LBP and CNN + GLCM, where the average accuracy increased as compared to LBP and GLCM alone. The fusion of CNN and morphological features was beneficial for the performance of cubic and quadratic SVMs, weighted KNN, and ANN. The boosted tree had the worst performance on both CNN and CNN + hand-crafted features when compared to other models. It follows that the boosted tree is not a good fit on the data having rich information and higher features dimension owing to its under-fitting nature (Ali et al., 2017).

As inferred from Table 2.4, the fusion CNN-extracted features with hand-crafted features did not lead to a significant improvement of accuracy as compared with CNN. This fusion did not improve but rather reduced the classification performance of ANN. A similar result has been obtained by (Wei Tan et al., 2018) who found that the ANN classifier with CNN features was the best method for the classification of plant species. However, it should be noted that the performance of a classifier is affected by the learning algorithm used and the features used for classification (Bakhshipour et al., 2018).

When comparing a range of feature extraction methods to classify corn seeds, (Yang et al., 2015) used hyperspectral imaging for the classification of four corn varieties. In their study, hyperspectral data was collected from germ and endosperm sides and recognition accuracy of 98.2% and 96.3%, respectively, was achieved by

Table 2.4: Classification accuracy (%) of trained classifiers on the testing data using various features.

Feature	**ANN**	**Cubic SVM**	**Quadratic SVM**	**Weighted kNN**	**Boosted Tree**	**Bagged Tree**	**Linear Discriminant**
Color	78.9	78.2	70.2	66.6	41.6	53.8	33.9
Morphological	45.4	45.5	43.8	44.3	39.9	44.2	30.1
LBP	85.7	84.7	84.1	75.7	64.4	76.3	49.5
GLCM	74.0	81.9	76.9	73.4	58.1	67.6	33.6
Color + Morph. + LBP + GLCM	72.0	72.0	73.7	53.6	50.9	69.4	45.5
CNN	**98.1**	93.8	93.2	91.7	81.7	89.1	91.3
CNN + Color	97.2	93.4	93.2	**93.8**	**87.4**	**93.6**	91.7
CNN + Morphological	88.6	82.8	82.1	62.6	28.6	24.3	20.3
CNN + LBP	93.8	**95.6**	**94.4**	91.3	81.9	92.0	**92.8**
CNN + GLCM	94.0	92.6	92.2	87.1	80.1	89.5	92.0
CNN + All	90.2	84.5	85.0	63.5	27.7	23.7	47.2

+

the SVM model. (Dong et al., 2018) used near-infrared spectroscopy to classify 3 types of corn. The highest classification accuracy of 80% in the study was achieved by the SVM model. Despite the high accuracy of the results, expensive devices such as multispectral imaging, hyperspectral imaging, and near-infrared spectroscopy were used mostly in related studies. In the proposed method, however, a low-cost digital camera was used. Thus, from an economic standpoint, a very cost-effective system was developed.

Besides, in most of these studies, the models were built based on the data related to the particular orientation of the sample corn seed that cannot be applied for industrial sorting systems (Wakholi et al., 2018). However, the proposed method is independent of the seed orientation, so classification could be performed in industrial settings with high accuracy. Because the CNN-ANN configuration had the best performance and accuracy, we presented cross-entropy performance and error histogram only for this case (Figure 2.4 and Figure 2.5). Cross-entropy is commonly used to describe the average error between calculated output and target output in the logarithmic scale. As visible in Figure 2.4, the best validation performance is obtained at a minimum cross-entropy error of 0.0028687 in 165 iterations. The validation accuracy of the validation set decreased continuously after the 165th epoch, but the training accuracy increased. This was an indication of overfitting and a reason to stop the training process at epoch 165. The outcome cross-entropy was almost zero indicating the superb performance of ANN. The plot indicates how the trained model fits the data set through maximum possible errors that can occur. It is worth noting that the errors are very close to zero. This proves that the proposed system performs the classification successfully with acceptable error.

The confusion matrices for all the seven diagnostic models based on the features for which they showed the best performance are presented in Figure 2.6, Figure 2.7. The rows in the confusion matrix correspond to the predicted class (Output Class) and the columns correspond to the true class (Target Class). The cells in the main diagonal of figures include the number of correctly classified samples, while other cells include the misclassified data. As shown in Figure 2.6 (a), the diagnostic model of ANN using CNN features had the best performance for most classes, except classes 3 and 6. In class 3, nine samples were classified wrongly as other classes and in class 6, there were 11 samples misclassified as other classes. Nevertheless, satisfying accuracies of 96.4% and 95.6% were achieved for classes 3

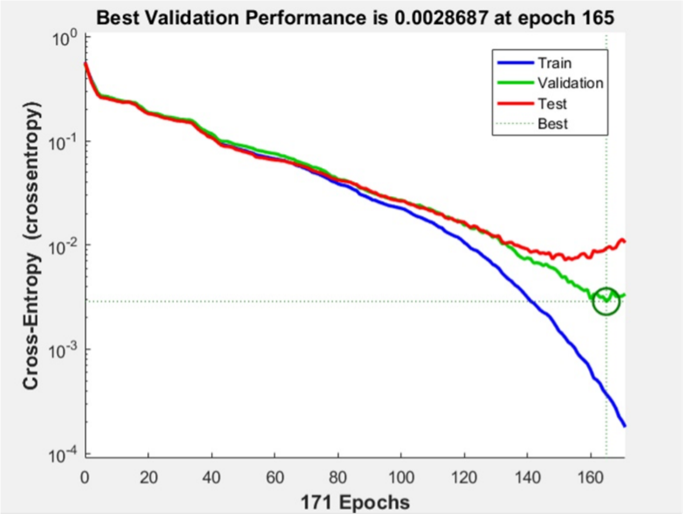


Figure 2.4: Cross-entropy versus the epochs plot for training, validation, and testing phases of the ANN-based on CNN features.

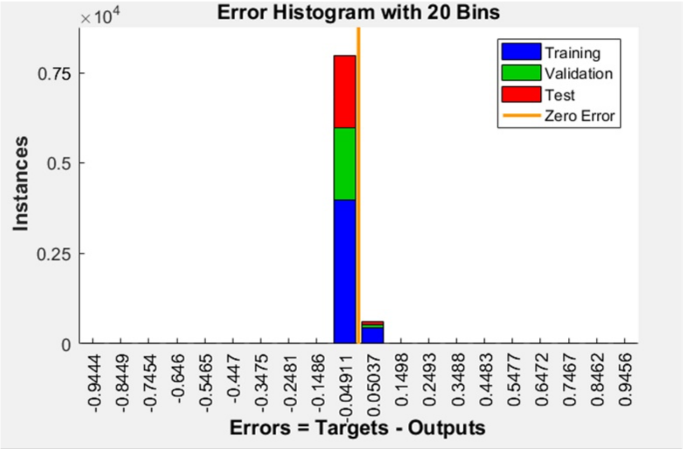
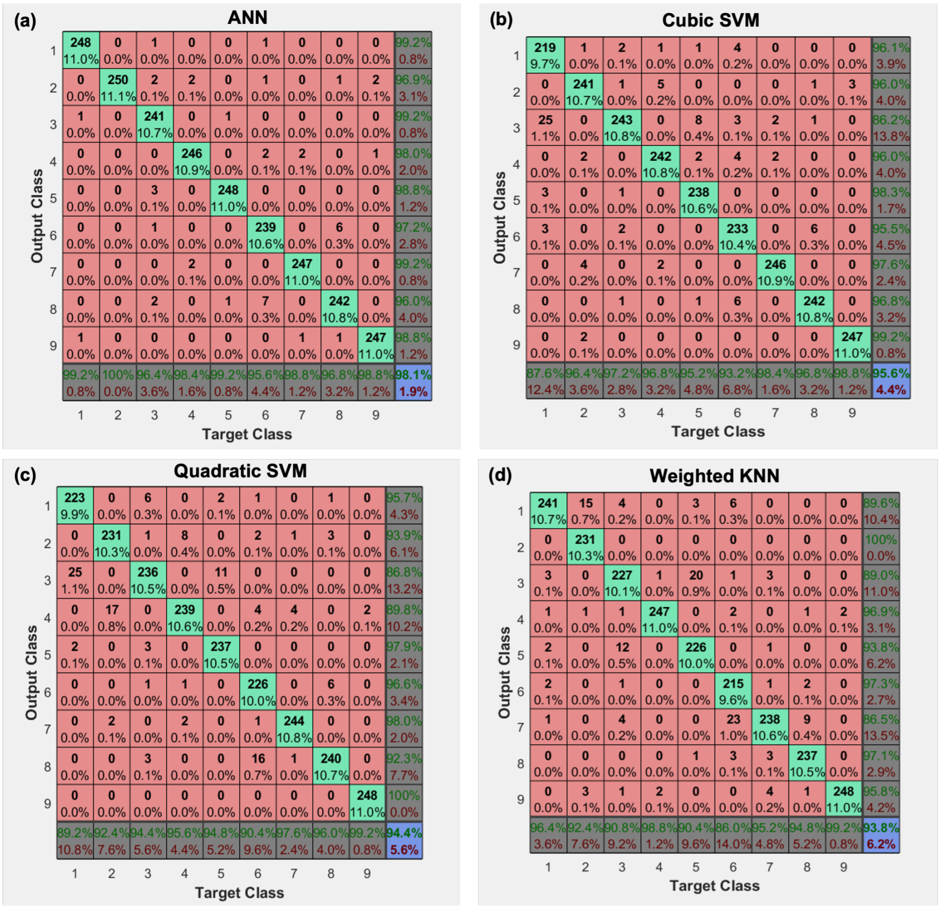


Figure 2.5: Error histogram for training, validation, and test sets for the ANN-based on CNN features.



(a) ANN

Output Class / Target Class

	1	2	3	4	5	6	7	8	9	
1	248 11.0%	0 0.0%	1 0.0%	0 0.0%	0 0.0%	1 0.0%	0 0.0%	0 0.0%	0 0.0%	99.2% 0.8%
2	0 0.0%	250 11.1%	2 0.1%	2 0.1%	0 0.0%	1 0.0%	0 0.0%	1 0.0%	2 0.1%	96.9% 3.1%
3	1 0.0%	0 0.0%	241 10.7%	0 0.0%	1 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	99.2% 0.8%
4	0 0.0%	0 0.0%	0 0.0%	246 10.9%	0 0.0%	2 0.1%	2 0.1%	0 0.0%	1 0.0%	98.0% 2.0%
5	0 0.0%	0 0.0%	3 0.1%	0 0.0%	248 11.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	98.8% 1.2%
6	0 0.0%	0 0.0%	1 0.0%	0 0.0%	0 0.0%	239 10.6%	0 0.0%	6 0.3%	0 0.0%	97.2% 2.8%
7	0 0.0%	0 0.0%	0 0.0%	2 0.1%	0 0.0%	0 0.0%	247 11.0%	0 0.0%	0 0.0%	99.2% 0.8%
8	0 0.0%	0 0.0%	2 0.1%	0 0.0%	1 0.0%	7 0.3%	0 0.0%	242 10.8%	0 0.0%	96.0% 4.0%
9	1 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	1 0.0%	1 0.0%	247 11.0%	98.8% 1.2%
	99.2% 0.8%	100% 0.0%	96.4% 3.6%	98.4% 1.6%	99.2% 0.8%	95.6% 4.4%	98.8% 1.2%	96.8% 3.2%	98.8% 1.2%	98.1% 1.9%

(b) Cubic SVM

Output Class / Target Class

	1	2	3	4	5	6	7	8	9	
1	219 9.7%	1 0.0%	2 0.1%	1 0.0%	1 0.0%	4 0.2%	0 0.0%	0 0.0%	0 0.0%	96.1% 3.9%
2	0 0.0%	241 10.7%	1 0.0%	5 0.2%	0 0.0%	0 0.0%	0 0.0%	1 0.0%	3 0.1%	96.0% 4.0%
3	25 1.1%	0 0.0%	243 10.8%	0 0.0%	8 0.4%	3 0.1%	2 0.1%	1 0.0%	0 0.0%	86.2% 13.8%
4	0 0.0%	2 0.1%	0 0.0%	242 10.8%	2 0.1%	4 0.2%	2 0.1%	0 0.0%	0 0.0%	96.0% 4.0%
5	3 0.1%	0 0.0%	1 0.0%	0 0.0%	238 10.6%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	98.3% 1.7%
6	3 0.1%	0 0.0%	2 0.1%	0 0.0%	0 0.0%	233 10.4%	0 0.0%	6 0.3%	0 0.0%	95.5% 4.5%
7	0 0.0%	4 0.2%	0 0.0%	2 0.1%	0 0.0%	0 0.0%	246 10.9%	0 0.0%	0 0.0%	97.6% 2.4%
8	0 0.0%	0 0.0%	1 0.0%	0 0.0%	1 0.0%	6 0.3%	0 0.0%	242 10.8%	0 0.0%	96.8% 3.2%
9	0 0.0%	2 0.1%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	247 11.0%	99.2% 0.8%
	87.6% 12.4%	96.4% 3.6%	97.2% 2.8%	96.8% 3.2%	95.2% 4.8%	93.2% 6.8%	98.4% 1.6%	96.8% 3.2%	98.8% 1.2%	95.6% 4.4%

(c) Quadratic SVM

Output Class / Target Class

	1	2	3	4	5	6	7	8	9	
1	223 9.9%	0 0.0%	6 0.3%	0 0.0%	2 0.1%	1 0.0%	0 0.0%	1 0.0%	0 0.0%	95.7% 4.3%
2	0 0.0%	231 10.3%	1 0.0%	8 0.4%	0 0.0%	2 0.1%	1 0.0%	3 0.1%	0 0.0%	93.9% 6.1%
3	25 1.1%	0 0.0%	236 10.5%	0 0.0%	11 0.5%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	86.8% 13.2%
4	0 0.0%	17 0.8%	0 0.0%	239 10.6%	0 0.0%	4 0.2%	4 0.2%	0 0.0%	2 0.1%	89.8% 10.2%
5	2 0.1%	0 0.0%	3 0.1%	0 0.0%	237 10.5%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	97.9% 2.1%
6	0 0.0%	0 0.0%	1 0.0%	1 0.0%	0 0.0%	226 10.0%	0 0.0%	6 0.3%	0 0.0%	96.6% 3.4%
7	0 0.0%	2 0.1%	0 0.0%	2 0.1%	0 0.0%	1 0.0%	244 10.8%	0 0.0%	0 0.0%	98.0% 2.0%
8	0 0.0%	0 0.0%	3 0.1%	0 0.0%	0 0.0%	16 0.7%	1 0.0%	240 10.7%	0 0.0%	92.3% 7.7%
9	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	248 11.0%	100% 0.0%
	89.2% 10.8%	92.4% 7.6%	94.4% 5.6%	95.6% 4.4%	94.8% 5.2%	90.4% 9.6%	97.6% 2.4%	96.0% 4.0%	99.2% 0.8%	94.4% 5.6%

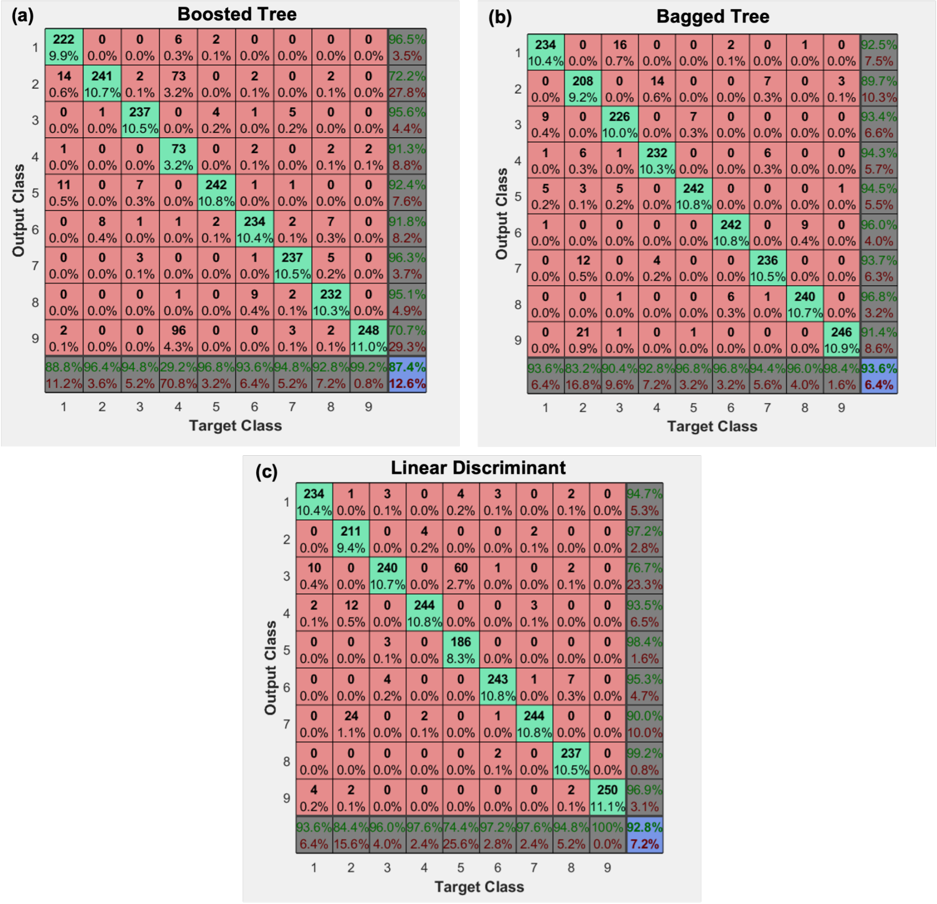
(d) Weighted KNN

Output Class / Target Class

	1	2	3	4	5	6	7	8	9	
1	241 10.7%	15 0.7%	4 0.2%	0 0.0%	3 0.1%	6 0.3%	0 0.0%	0 0.0%	0 0.0%	89.6% 10.4%
2	0 0.0%	231 10.3%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
3	3 0.1%	0 0.0%	227 10.1%	1 0.0%	20 0.9%	1 0.0%	3 0.1%	0 0.0%	0 0.0%	89.0% 11.0%
4	1 0.0%	1 0.0%	0 0.0%	247 11.0%	0 0.0%	2 0.1%	0 0.0%	1 0.0%	2 0.1%	96.9% 3.1%
5	2 0.1%	0 0.0%	12 0.5%	0 0.0%	226 10.0%	0 0.0%	1 0.0%	0 0.0%	0 0.0%	93.8% 6.2%
6	2 0.1%	0 0.0%	1 0.0%	0 0.0%	0 0.0%	215 9.6%	1 0.0%	2 0.1%	0 0.0%	97.3% 2.7%
7	1 0.0%	0 0.0%	4 0.2%	0 0.0%	0 0.0%	23 1.0%	238 10.6%	9 0.4%	0 0.0%	86.5% 13.5%
8	0 0.0%	0 0.0%	0 0.0%	0 0.0%	1 0.0%	3 0.1%	3 0.1%	237 10.5%	0 0.0%	97.1% 2.9%
9	0 0.0%	3 0.1%	1 0.0%	2 0.1%	0 0.0%	0 0.0%	4 0.2%	1 0.0%	248 11.0%	95.8% 4.2%
	96.4% 3.6%	92.4% 7.6%	90.8% 9.2%	98.8% 1.2%	90.4% 9.6%	86.0% 14.0%	95.2% 4.8%	94.8% 5.2%	99.2% 0.8%	93.8% 6.2%

Figure 2.6: The confusion matrices of the different classifiers on the testing data set based on the features that had the best performance as described in Table 4: (a) CNN; (b) CNN + LBP; (c) CNN + LBP; and (d) CNN + color. Classes are: 1 = KSC 201, 2 = KSC 704, 3 = KSC 290, 4 = KSC 380, 5 = KSC 301, 6 = KSC 400, 7 = KSC 260, 8 = KSC 647, and 9 = KSC 410.

and 6, respectively, where other classes had an accuracy between 97% and 100%. The experimental results demonstrate that the CNN-ANN classifier achieved superior performance in all metrics compared to other studied frameworks, where the average accuracy, precision, recall, and F1-score were 98.1%, 98.2%, 98.1%, and 98.1%, respectively. An obvious conclusion could be drawn from Figure 2.6(b) that in the cubic SVM model based on the fusion of CNN and LBP features it was 99 misclassifications, while for the ANN model trained with CNN features only 42 corn



(a) Boosted Tree

	1	2	3	4	5	6	7	8	9	
1	222 9.9%	0 0.0%	0 0.0%	6 0.3%	2 0.1%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	96.5% 3.5%
2	14 0.6%	241 10.7%	2 0.1%	73 3.2%	0 0.0%	2 0.1%	0 0.0%	2 0.1%	0 0.0%	72.2% 27.8%
3	0 0.0%	1 0.0%	237 10.5%	0 0.0%	4 0.2%	1 0.0%	5 0.2%	0 0.0%	0 0.0%	95.6% 4.4%
4	1 0.0%	0 0.0%	0 0.0%	73 3.2%	0 0.0%	2 0.1%	0 0.0%	2 0.1%	2 0.1%	91.3% 8.8%
5	11 0.5%	0 0.0%	7 0.3%	0 0.0%	242 10.8%	1 0.0%	1 0.0%	0 0.0%	0 0.0%	92.4% 7.6%
6	0 0.0%	8 0.4%	1 0.0%	1 0.0%	2 0.1%	234 10.4%	2 0.1%	7 0.3%	0 0.0%	91.8% 8.2%
7	0 0.0%	0 0.0%	3 0.1%	0 0.0%	0 0.0%	1 0.0%	237 10.5%	5 0.2%	0 0.0%	96.3% 3.7%
8	0 0.0%	0 0.0%	0 0.0%	1 0.0%	0 0.0%	9 0.4%	2 0.1%	232 10.3%	0 0.0%	95.1% 4.9%
9	2 0.1%	0 0.0%	0 0.0%	96 4.3%	0 0.0%	0 0.0%	3 0.1%	2 0.1%	248 11.0%	70.7% 29.3%
	88.8% 11.2%	96.4% 3.6%	94.8% 5.2%	29.2% 70.8%	96.8% 3.2%	93.6% 6.4%	94.8% 5.2%	92.8% 7.2%	99.2% 0.8%	87.4% 12.6%

(b) Bagged Tree

	1	2	3	4	5	6	7	8	9	
1	234 10.4%	0 0.0%	16 0.7%	0 0.0%	0 0.0%	2 0.1%	0 0.0%	1 0.0%	0 0.0%	92.5% 7.5%
2	0 0.0%	208 9.2%	0 0.0%	14 0.6%	0 0.0%	0 0.0%	7 0.3%	0 0.0%	3 0.1%	89.7% 10.3%
3	9 0.4%	0 0.0%	226 10.0%	0 0.0%	7 0.3%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	93.4% 6.6%
4	1 0.0%	6 0.3%	1 0.0%	232 10.3%	0 0.0%	0 0.0%	6 0.3%	0 0.0%	0 0.0%	94.3% 5.7%
5	5 0.2%	3 0.1%	5 0.2%	0 0.0%	242 10.8%	0 0.0%	0 0.0%	0 0.0%	1 0.0%	94.5% 5.5%
6	1 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	242 10.8%	0 0.0%	9 0.4%	0 0.0%	96.0% 4.0%
7	0 0.0%	12 0.5%	0 0.0%	4 0.2%	0 0.0%	0 0.0%	236 10.5%	0 0.0%	0 0.0%	93.7% 6.3%
8	0 0.0%	0 0.0%	1 0.0%	0 0.0%	0 0.0%	6 0.3%	1 0.0%	240 10.7%	0 0.0%	96.8% 3.2%
9	0 0.0%	21 0.9%	1 0.0%	0 0.0%	1 0.0%	0 0.0%	0 0.0%	0 0.0%	246 10.9%	91.4% 8.6%
	93.6% 6.4%	83.2% 16.8%	90.4% 9.6%	92.8% 7.2%	96.8% 3.2%	96.8% 3.2%	94.4% 5.6%	96.0% 4.0%	98.4% 1.6%	93.6% 6.4%

(c) Linear Discriminant

	1	2	3	4	5	6	7	8	9	
1	234 10.4%	1 0.0%	3 0.1%	0 0.0%	4 0.2%	3 0.1%	0 0.0%	2 0.1%	0 0.0%	94.7% 5.3%
2	0 0.0%	211 9.4%	0 0.0%	4 0.2%	0 0.0%	0 0.0%	2 0.1%	0 0.0%	0 0.0%	97.2% 2.8%
3	10 0.4%	0 0.0%	240 10.7%	0 0.0%	60 2.7%	1 0.0%	0 0.0%	2 0.1%	0 0.0%	76.7% 23.3%
4	2 0.1%	12 0.5%	0 0.0%	244 10.8%	0 0.0%	0 0.0%	3 0.1%	0 0.0%	0 0.0%	93.5% 6.5%
5	0 0.0%	0 0.0%	3 0.1%	0 0.0%	186 8.3%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	98.4% 1.6%
6	0 0.0%	0 0.0%	4 0.2%	0 0.0%	0 0.0%	243 10.8%	1 0.0%	7 0.3%	0 0.0%	95.3% 4.7%
7	0 0.0%	24 1.1%	0 0.0%	2 0.1%	0 0.0%	1 0.0%	244 10.8%	0 0.0%	0 0.0%	90.0% 10.0%
8	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	2 0.1%	0 0.0%	237 10.5%	0 0.0%	99.2% 0.8%
9	4 0.2%	2 0.1%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	2 0.1%	250 11.1%	96.9% 3.1%
	93.6% 6.4%	84.4% 15.6%	96.0% 4.0%	97.6% 2.4%	74.4% 25.6%	97.2% 2.8%	97.6% 2.4%	94.8% 5.2%	100% 0.0%	92.8% 7.2%

Figure 2.7: The confusion matrices of the different classifiers on the testing data set based on the features that had the best performance as described in Table 4: (a) CNN + color; (b) CNN + color; and (c) CNN + LBP. Here 1, 2, 3, 4, 5, 6, 7, 8, and 9 depict KSC 201, KSC 704, KSC 290, KSC 380, KSC 301, KSC 400, KSC 260, KSC 647, and KSC 410.

seeds were misclassified. Among nine classes, the lowest (87.6%) and the highest (98.8%) diagnosis rates belonged to classes 1 and 9, respectively. The average accuracy, precision, recall, and F1-score of cubic SVM with CNN + LBP features were 95.6%, 95.7%, 95.6%, and 95.7%, respectively. The confusion matrix in Figure 2.6(c) shows the experimental results of the quadratic SVM classifier using the combination of CNN and LBP features. By applying this configuration, 126 out of 2250 corn samples were misclassified. A similar result was obtained by cubic SVM classifier using the same combination of CNN + LBP features, where the lowest and highest classification accuracies were observed for classes 1 and 9, respectively. Additionally, in both systems, 25 samples from class 1 were classified as class 3 which had the highest volume of misclassified samples among different classes. The average accuracy, precision, recall, and F1-score of this configuration were 94.4%, 94.6%, 94.4%, and 94.5%, respectively. The experimental results of the weighted KNN based CNN + color classifier is shown in Figure 2.6(d). For this system, the minimum and maximum classification accuracies were observed in classes 6 and 9 (86.0% and 99.2%, respectively). Also, the most obvious misclassifications occurred in 6-vs-7 and 5-vs-3. The average amount of the performance measures of the weighted KNN based CNN + color classifier including accuracy, precision, recall, and F1-score were obtained 93.8%, 94.0%, 93.8%, and 93.9%, respectively.

For confusion matrix of boosted tree-based CNN + color model (Figure 2.7a), class 4 with the lowest accuracy (29.2%) and class 9 with the highest accuracy (99.2%) were classified, whereas all of the other classes were classified with accuracies between approximately 89 to 97%. This classifier had serious problems in discriminating between classes 4 and 9 and also between 4 and 2. The average accuracy, precision, recall, and F1-score of the boosted tree-based CNN + color model on the test data set were 87.4%, 89.1%, 87.4%, and 88.2%, respectively. The confusion matrix of the bagged tree model based on the fusion of CNN and color features is shown in Figure 2.7(b). We can see that class 2 had the worst classification error among all classes since a significant amount of the corns (33 samples) were confused with classes 7 and 9. Class 9 had better diagnostic accuracy in comparison with other classes. The mean values of all performance measures for the bagged tree-based CNN + color model were equal to 93.6%. In Figure 2.7(c) the performance of the LDA-based CNN + LBP classifier is demonstrated. An important interpretation of this plot is that the diagnosis accuracy of classes 5 and 2 was lower than 85% while it was 93% for other classes. The highest number of misclassifications occurred between classes 5 and 3 (60 samples), followed by 2 and 4 (24 samples). The average values of accuracy, precision, recall, and F1-score of LDA-based CNN + LBP classifier were 92.8%, 93.6%, 92.8%, and 93.2%, respectively.

The classification time of different models used is shown in Table 2.5. Classification time indicates the time required to classify 2250 seeds in the test set, using a trained algorithm. As could be seen from Table 2.5, the classification time required for all classifiers was reasonable since a single seed could be classified in less than 0.05 seconds.

When individual hand-crafted features were used to classify the classes, a weighted KNN classifier based on color features required the shortest classification time among all algorithms. In contrast, a cubic SVM classifier based on LBP features required the longest classification time. In the same Table, it becomes clear that computational time increased for classifiers that used a fusion of hand-crafted features compared to classifiers that used individual hand-crafted features. Classification time for all classifiers using only CNN features and fusion of CNN-extracted with hand-crafted features was relatively longer than for classifiers based only on hand-crafted features. The longest classification time was observed for cases based on the fusion of all features. Cubic SVM was more time-consuming than other classifiers. The bolded values show the least classification time achieved by each classifier.

Although the diagnosis time for the CNN-ANN classifier was longer than some of the other models, the required time for classification of a single corn image (0.01 s) was acceptable. It should be noted that the classification time of algorithms is dependent on hardware resources. Therefore, using the next generation of GPUs can ensure a shorter classification time for the proposed method in this study.

2.5 Conclusions

Based on the present research, the following conclusions can be drawn:

- Classifying corn seed varieties in the batch is a challenging process because of high inter-class similarity. To resolve this problem, a novel technique based on deep learning of computer images, using the combination of hand-crafted and non-handcrafted (CNN-extracted) features was proposed. The application of CNN-extracted features in combination with hand-crafted features improved the accuracy of nine corn seed varieties classification as compared to hand-crafted features alone. For example, the biggest improvement was observed with the Linear Discriminant algorithm, where the accuracy of the model increased from 33.6 to 92%, while the ANN algorithm improved the accuracy of classification from 85.7 to 93.8%.

Table 2.5: The classification time (s) of different classifiers, based on different features, on the testing data.

Feature	**ANN**	**Cubic SVM**	**Quadratic SVM**	**Weighted kNN**	**Boosted Tree**	**Bagged Tree**	**Linear Discriminant**
Color	8.4	25.9	12.1	6.1	8.2	12.0	9.8
Morphological	12.0	27.2	22.4	10.6	10.2	16.1	11.3
LBP	14.9	32.9	24.9	13.3	12.1	17.5	13.3
GLCM	18.2	34.9	28.8	8.7	9.3	15.3	10.9
Color + Morphological + LBP + GLCM	19.3	37.0	28.5	21.3	20.7	22.4	17.5
CNN	26.8	39.5	38.4	28.2	25.6	24.5	20.2
CNN + Color	24.5	41.8	31.2	14.2	16.0	21.6	22.1
CNN + Morphological	28.6	38.0	32.0	16.2	17.4	20.7	19.5
CNN + LBP	32.4	45.6	34.3	17.6	18.0	20.9	18.6
CNN + GLCM	21.9	38.0	29.6	18.7	15.1	21.3	23.0
CNN + Color + Morphological + LBP + GLCM	47.5	52.2	47.5	72.0	95.4	109.9	89.8

- In our experiments, hand-crafted features included 59 local binary pattern texture features, 17 morphological features, 6 color features, and 5 gray-level co-occurrence matrix features. Non-handcrafted features have been extracted by feeding images to a VGG-16 pre-trained CNN model. Both hand-crafted and non-handcrafted features were used to train different classifiers including ANN, cubic SVM, quadratic SVM, weighted KNN, boosted tree, bagged tree, and LDA. The best accuracy of 98% was achieved by concatenating CNN-extracted features with ANN.

- Although the fusion of CNN-extracted features added time to the classification process, the proposed CNN-ANN algorithm had an acceptable classification time of 26.8 s. 4. Our results demonstrated that CNN-ANN is a promising architecture for the accurate classification of corn varieties with the potential for further improvement. As the next step, we are going to explore the effect of dimensional reduction on classification time and accuracy.