Measurement 173 (2021) 108650

Contents lists available at ScienceDirect

Measurement

journal homepage: www.elsevier.com/locate/measurement

A deep learning approach to measure stress level in plants due to Nitrogen deficiency

Shiva Azimi, Taranjit Kaur, Tapan K. Gandhi *

Department of Electrical Engineering, Indian Institute of Technology-Delhi, Hauz Khas, New Delhi 110016, India

ARTICLE INFO

Keywords:
Computer vision
Deep learning
Nitrogen stress
Plant phenotyping

ABSTRACT

Stress due to nutrients deficiency in plants can reduce the agricultural yield significantly. Nitrogen, an essential nutrient, is a crucial growth-limiting factor and is the prime component of amino acids, proteins, nucleic acids, and chlorophyll. Nitrogen deficiency affects certain visible plant traits such as area, color, the number of leaves and plant height, etc. With the recent advancements in imaging technology, computer vision-based plant phenomics has become a promising field of plant research and management. Such imaging-based techniques are non-destructive and much faster with higher levels of automation. In this work, we have proposed an automatic image-based plant phenotyping approach for stress classification in plant shoot images. In this proposed phenotyping approach, a 23-layered deep learning technique is proposed and compared with traditional Machine Learning techniques and few other deep architectures. Results reveal that a simple 23-layered deep learning architecture is comparable to the established state of art deep learning architectures like ResNet18 and NasNet Large (having millions of trainable parameters) in yielding ceiling level stress classification from plant shoot images. In addition, the proposed model also outperforms traditional Machine Learning techniques by achieving an average of 8.25% better accuracy.

1. Introduction

The world is expected to touch a population of 8.6 billion by 2030, and thus, being able to cultivate enough crops plays a crucial role in managing food and health security [1]. To feed the world's rapidly growing population, the process of agriculture needs to be evolved for precision farming. This method of farming employs a wide variety of Internet of Things (IoT) sensors that measure soil characteristics and imaging devices that keep track of certain plant traits like color, size, shape, etc. Being able to study these observable traits of a plant using different sensors and imaging devices plays a vital role in this transformation and is known as plant phenomics [2].

Using traits of plants for detecting physiological changes in response to abiotic stresses such as lack of water, nutrients, poor lightning condition, etc., has a significant role in plant and crop management. After moisture induced stress, nutrient induced stress in plants is the most critical factor and can significantly reduce the agricultural yield [3]. Nutrients induced stress can result from either low level or excess level of minerals in the plant system. Understanding and measuring nutrient induced stress is a complicated process as various aspects such as soil science, plant physiology, biochemistry, agronomy, and molecular biology need to be considered. The early detection and quantification of stress levels in plants has an important role in the final yield. Nitrogen is one of the most important nutrient required by plants and is a crucial growth-limiting factor as it is an essential component in amino acids, proteins, nucleic acids, and chlorophyll. Nitrogen deficiency can results in disturbed root–shoot ratio, short lateral branches, small leaves, Chloroplast disintegration, and even death. Different levels of nitrogen deficiency effect certain visually accessible plant traits such as leaf color, leaf area, number of leaves, and plant height.

Most traditional phenotyping techniques are destructive and time-consuming with intensive labor requirements, and hence, smart automated plant phenotyping techniques that can help identify the stress level in less time without disturbing the plant are needed. With the recent advances in computer vision and object recognition, it has become quite straightforward to capture high resolution images in both the visible as well as the infrared spectrum [4,5]. Thanks to these developments, image-based plant phenotyping can now be done on a small scale (controlled laboratory), in the greenhouse, or in the field [6–8], and has become a promising field of plant research and management. Such imaging-based techniques are not only non-destructive but are also much faster, with high levels of automation. However, Using high-throughput imaging and computer vision methods for plant phenotyping presents challenges such as data acquisition and analysis

* Corresponding author.
E-mail address: tgandhi@ee.iitd.ac.in (T.K. Gandhi).

https://doi.org/10.1016/j.measurement.2020.108650
Received 23 June 2020; Received in revised form 28 September 2020; Accepted 20 October 2020
Available online 27 October 2020

that require an expertise in the fields of biology, engineering, and mathematics [9].

Machine Learning (ML) methods offer many advantages in the analysis of big data in different fields of research, such as healthcare,industries, agriculture, etc. However, in agriculture, obtaining stress information using images is challenging due to lack of texture in multivariate plant images, occlusion, complicated and invisible part of plants [10,11]. Out of various ML techniques available at our disposal, Deep Learning (DL) method that uses different convolutions to presents the data hierarchically is gaining ground [12].

The raw images that arise in imaging based phenotyping applications generally contain too much information for a traditional ML techniques to work efficiently. To tackle this, plenty of work in traditional ML research has focused on pre-computation of domain-specific image keypoints and features and then train a classifier in the corresponding representation space [13–17]. Some of the popular classification methods that have been used in plant phenomics application include Support Vector Machines (SVM) [18], Decision trees (DT) [19], and K-Nearest Neighbors (KNN) [20]. For example, the use of a linear SVM classifier for assessing water stress in the early growth stages of maize has been considered in [21], while the authors in [22] used SVM on the extracted features for a hierarchical classification of soybean plants. The main limitation of these traditional ML techniques is the fact that the handcrafted features suit a particular kind of data, and thus, applications of the algorithm are limited to specific tasks. This means that an approach that works well for one task may fail to perform for a different task. This kind of specificity associated with these traditional approaches severely limits the applicability in situations that demand a certain level of generality and has motivated researchers to look for methods that are quite generic in terms of applicability. This has further strengthened the focus on CNN, which is a much general ML approach.

Convolutional Neural Network (CNN) is a DL method that specializes in image recognition and has been used in variety of computer vision problems across different fields [23]. CNN's have additional feature-detecting convolution layers along with an artificial neural networks (ANNs) layer that does the classification [24]. CNN's have been readily accepted by the computer vision community and have been applied successfully in fields such as life sciences, medicine [25] and agriculture [26]. CNN has been widely used for the classification of plants and leaves in agriculture [27–29]. In [28], the authors have represented leaf data by defining a way to quantify the features and trained a CNN based on raw leaf data. This was followed by using a Deconvolutional Network (DN) approach to understand how the CNN specifies the leaf data. Their results showed the strength of detecting features using CNN as compared to hand-crafted features. DL techniques have also been used in applications such as counting the number of seeds per pod for soybeans [30], counting the number of wheat ear under field conditions [31], plant identification [29,32], identification of plant diseases [33–37], etc. Some works have also concentrated on finding out future parameters, including corn produce [38], amount of soil moisture in the field [39], and weather [40]. An et al. [41] has proposed a deep convolutional neural network (DCNN) for classification and identification of maize drought stress in the field. They achieved 98.14 percent classification accuracy for drought stress. In this paper, we have considered automated phenotyping of stress level classification from Sorghum plant shoots due to nitrogen deficiency using a 23-layered CNN architecture. Compared to the existing works, the main contributions of this paper are as follows:

- We propose to use a 23-layered CNN architecture unlike other works existing in the literature that mostly use classical ML techniques for classification of nitrogen deficiency in plants.
- We also provide a thorough performance analysis of the classical ML techniques using combinations of different keypoint detectors and descriptors.
- A performance comparison of the DL model with reference to the traditional ML methods and some of the well established pre-trained DCNN models is also presented.

Interestingly, the computer vision based DL classification method for early and fast detection of plant nitrogen content (as discussed in our work) has many envisioned applications apart from increasing the crop yield as summarized below:

- As we are suggesting the use of complete shoot images, our method is independent of the lighting conditions, the time of the day when the images are taken, and has the potential for a high degree of automation.
- Such an identification system would be beneficial to the potential farmers and researchers for timely intervention and mitigation of the problems by applying the proper crop management strategies that can effectively boost the crop yields.
- The image based nitrogen stress classification methods can not only detect the lack of plant nitrogen, but also the access of it. This helps in the optimization of nitrogen fertilization. This can help farmers to avoid over fertilization which affects optimum plant productivity, causing unnecessary expenditure on the part of farmers, and negatively impacts the soil health.
- These methods can be useful not only for the people who are into farming, but also for people who are going to consume the yield, as the high-nitrate diet is an alarming factor in the development of several human diseases such as methahemoglobinaemia, gastric and bladder cancer. The approaches discussed in our paper can be employed to detect the nitrogen content in the plants just before the harvest and help in deciding whether the nitrogen levels are in the safe range for human consumption.
- Several research works have been proposed to detect and classify plant stresses using image processing and Machine Learning techniques. These approaches have attempted to build image classifiers using the handcrafted shape, color, and texture features extracted from the individual crop leaf images. The classifier dependency on handcrafted features causes a lack of automation. To overcome this, the present work proposes a computationally simple deep convolutional neural networks (DCNN) architecture that has shown impressive results for nitrogen stress classification without using any handcrafted features. This brings the desire of adopting deep learning in computer vision to develop a comprehensive nitrogen stress recognition system which could be used as an effective tool in constructing high precision crop management strategies.

The remainder of the paper is structured as follows. Section 2 describes the dataset and the proposed methodology. Section 3 provides the results and discussions. Finally, in Section 4, we provide some perspectives and conclude.

2. Materials and methods

This section gives an overview of the dataset and methodology that we employ for finding the nitrogen stress level in a Sorghum plant from the shoot images. At first, we explain the dataset used in our experiments, after that, we briefly discuss the image preprocessing stage that segments the plant shoot from other objects present in the image. Later, we will discuss about the conventional ML algorithms like SVM, DT, KNN, followed by the DL approach using CNN. The conventional ML techniques will work on a chosen set of attributes extracted using standard feature extraction algorithms like Scale-Invariant feature transform (SIFT) [42,43] and Histogram of oriented gradients (HOG) [44], etc.

(a) 0 degree (b) 90 degree

Fig. 1. Visualization of a sample in 0 degree (right) and 90 degree (left) from dataset.

Table 1
Number of images in each class.

Class	Number of images	
	0 degree	90 degree
100%	3408	3408
50%	5112	5112
10%	5114	5114

2.1. Dataset

The plants shoot images for this work have been obtained from a standard publicly available dataset containing almost 96,867 images of Sorghum plant shoot, provided by the Donald Danforth Plant Science Center (https://plantcv.//danforthcenter.org/pages/data-sets/sorghum_abiotice_stress.html) [45]. The dataset was prepared by subjecting 30 genetically diverse genotypes of Sorghum to certain levels of nitrogen stress. The motivation behind choosing Sorghum for the experiment is its ability to grow in relatively dry soil with limited inputs. Sorghum has many uses. It can be used for medical purposes, feeding animals and humans, fuel production, etc. Next to water stress, nitrogen stress is the most important environmental factor affecting the plant's yield. In this experiment, the phenotypic impacts of nitrogen stress on Sorghum were examined for three weeks employing high-throughput image based phenotyping. Three sets of experimental conditions were created based on the amount of nitrogen available to plants: the 100/100 treatment group, the 50/10 treatment group, and the 10/10 treatment group. In the first setup, 100% of the nitrogen requirement was available (100% ammonium/100% nitrate), in second setup 50% of the nitrogen requirement was available (50% ammonium/10% nitrate), and in third, 10% of the nitrogen requirement was available (10% ammonium/10% nitrate) [45].

Based on the amount of nitrogen availability, we have three classes of samples in this dataset. Plants with 100% as healthy (AA), 50% as semi stressed (AB), and 10% as severely stressed (AC). The original dataset includes both RGB and Near Infrared (NIR) images. For each plant, one top view and two sides view RGB and NIR images corresponding to 0° degree and 90° degree are taken. In this work we have used two sides view RGB images of the plant shoot for finding the level of nitrogen stress in the plant. The motivation behind using the RGB images being the fact that RGB cameras are cheaper and widely available, given the popularity of devices such as consumer cameras and smart-phones. One of the biggest challenge in using the dataset is taking care of difference in the age of plants that are captured over a period of 26 days after seedling of plants in each class. Table 1 gives the information about number of images per class. Fig. 1 shows some sample images from this dataset.

2.2. Image pre-processing

The first stage in pre-processing is to segment the plant shoot from the background. Images considered for this study were captured in the laboratory settings with white background. The notion behind the image pre-processing or background subtraction is to reduce the size of the feature vector. Without background in the images, we are able to reduce the number of keypoints significantly and subsequently this helps in reducing the computational cost. As shown in Fig. 2, we removed the plant to create the background using image patching technique through GNU Image Manipulation Program (GIMP) software. This background image contains the imaging setup, including the pot. For the ease of computation, the areas having imaging setup including pot were cropped. Once we had the cropped images, computational methods were employed to generate both intensity as well as color maps simultaneously. Finally, both the maps were merged to obtain the final mask for segmentation of the plant shoot.

In the first method for mask generation, the background image is used to detect the image pixels that belong to the plant. Here, both the image and the background were first converted to gray-scale and then binarized. The binarized background was subtracted from the image to give the mask. This mask was smoothed out by removing small, discrete areas, giving us the intensity based mask.

In the second method the color information in the image is used to generate mask, as the image pixels belonging to the plant are usually green in color. Here, first the image was converted from RGB to Lab color space, which is defined by a luminosity layer 'L', a chromaticity layer 'a' indicating where the color lies along the red–green axis, and chromaticity-layer 'b' indicating where the color lies along the blue–yellow axis. The entire color information is contained within the 'a' and 'b' layers. We then used k-mean clustering [46] on these layers to do color based segmentation. Segmentation of the green color yielded the required color based mask.

The final mask was then obtained by merging these two masks, and removing the outliers using the Cook's distance [47]. A white pixel in the mask, with Cook's distance larger than three times the mean Cook's distance was considered as an outliers and is removed.

The final mask was then applied to the cropped RGB image to segment the plant shoot. An overview of the process is shown in Fig. 2.

2.3. Classical ML approaches

Classical ML techniques are used widely in computer vision problems even today. These techniques basically involve three steps: Extracting descriptors for the detected keypoints, encoding these descriptors, and then training a classifier using the encoded descriptors (model creation). The details of the methods used in this work are provided in the following sub-sections.

2.3.1. Feature extraction

The first step in using the classical ML techniques involve feature extraction. Feature extraction involves detecting the keypoints and then describing the local patches in the neighborhood of these points. In the context of computer vision, points rich in information content are known as keypoints. These keypoints play a significant role in increasing/decreasing the computational cost depending on the number of data points. In the present work, we have used two of the most prominent image descriptors i.e., SIFT [42], and HOG [44] features. These features are extracted separately from plant images available in the dataset. Additionally, the performance of chosen descriptors is evaluated using different classifiers and the results are given in Section 3.

2.3.1.1. Scale-invariant feature transform (SIFT). In this work, we used SIFT descriptor for feature extraction from the segmented plant images [42,43]. The SIFT keypoint detector first generates a Difference-of-Gaussians (DoG) scale-space and then finds local extrema in that space. SIFT proceeds by first sampling the data regularly in space to approximate a density function. This density function has been used to generate a scale space and finally, a search is performed to find the local maxima in Hessian matrix. The subsequent section details the process of creation of scale space and the local maxima. The scale space

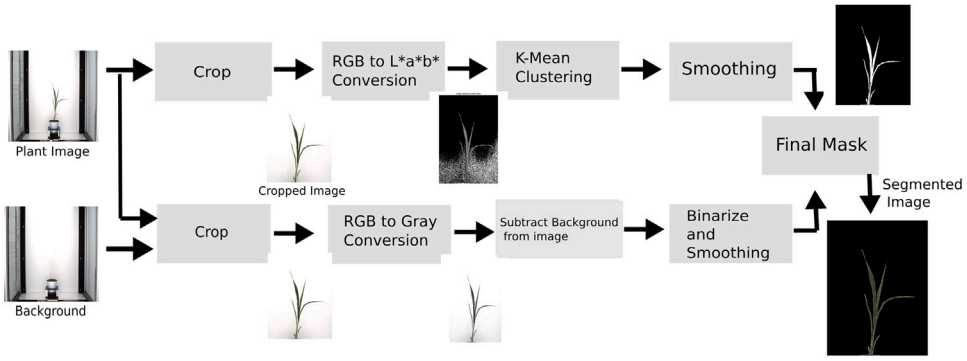


Fig. 2. Illustration of the image segmentation steps. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

is obtained by convolving the input image $I(x, y)$ with a number of Gaussian filters $G(x, y, \sigma)$ which have standard deviations σ that differ from one another by a fixed factor

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y), \tag{1}$$

Where $*$ represents the convolution operator. The SIFT keypoints are located at the extrema of the scale-space of the DoG function $D(x, y, \sigma)$ which is given as the difference between two images, whose scales vary by a factor k, i.e.,

$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma) \tag{2}$$

Given $D(x, y, \sigma)$, its local maxima and minima are obtained by comparing each of the points with their 8 neighbors that are at the same scale, and 9 neighbors that are up and down one scale. If this point is the minima or the maxima, then it is chosen as an extrema. In the next stage, the Laplacian is calculated for these extrema. The extremum location $\mathbf{Z}$, is

$$\mathbf{Z} = -\frac{\partial^2 D^{-1}}{\partial \mathbf{x}^2} \frac{\partial D}{\partial \mathbf{x}}. \tag{3}$$

The value at $\mathbf{Z}$ is compared to a threshold and if it is below it, the point $\mathbf{Z}$ is discarded. This gets rid of extrema having low contrast. SIFT descriptors are computed based on the gradient magnitude as well as orientation at each image point in the neighborhood of a keypoint, weighted by a Gaussian window as shown in (1). This step assigns a consistent orientation to the keypoints. The gradient magnitude m are then obtained as

$$m(x, y) = \sqrt{(L(x+1, y) - L(x-1, y))^2 + (L(x, y+1) - L(x, y-1))^2}. \tag{4}$$

The corresponding orientation θ is

$$\theta(x, y) = \arctan\left(\frac{L(x, y+1) - L(x, y-1)}{L(x+1, y) - L(x-1, y)}\right). \tag{5}$$

To obtain the descriptor, a set of orientation histograms are created using both magnitude and orientation values of image sample points in a neighborhood of the keypoint. The coordinates along with the gradient orientations are then rotated with respect to the key-point orientation, giving rotation invariance.

2.3.1.2. Histogram of oriented gradients (HOG). Another descriptor used for feature extraction from the segmented plant images is HOG [44]. HOG uses a window to generate a descriptor that is local to a detected image keypoint. The window, consisting of a regular square grid ($n \times n$) is centered upon the keypoint under consideration, and for each cell within the grid, a frequency histogram is generated to depict the edge orientations distribution. The gradient magnitude M and orientation θ is calculated as

$$M(x, y) = \left(g_x^2 + g_y^2\right)^{\frac{1}{2}}, \tag{6}$$

and

$$\theta(x, y) = \arctan\left(\frac{g_y}{g_x}\right), \tag{7}$$

Where g_x and g_y are gradient for each pixel of image I along x and y directions, respectively,

$$g_x = \frac{\partial I}{\partial x} = f(x+1, y) - f(x-1, y) \tag{8}$$

and

$$g_y = \frac{\partial I}{\partial y} = f(x, y+1) - f(x, y-1) \tag{9}$$

This is then followed by the quantization of the edge orientations into q bins. A vector $(q - D)$ is formed by concatenating the histogram counts for each cell. These vectors are further concatenated giving a $qn^2 - D$ vector for the entire window. Some implementations sample several windows in a non-overlapping $w * w$ grid in the key-point locality and concatenate all these windows to produce the final descriptor.

2.3.1.3. Fisher vector (FV). Once the SIFT and HOG descriptors are extracted for the plant images separately, the next step is to use them in the classification stage. However, since the rows of the descriptors are basically the feature vectors for each of the image keypoints, the size of the descriptors varies from image to image, as the number of keypoints detected for each image are different. This necessitates the quantization of the descriptors into a single vector of uniform size. This is called encoding, and in this work, we have used Fisher Vector (FV) [48] encoding.

A 'FV' is a statistical representation of the distribution of a set of vectors (image features in our case). The FV performs the encoding by creating a dictionary of k words using a Gaussian Mixture Model (GMM) for the descriptors. Once ready, the dictionary is used to encode the gradients of the log-likelihood of the features under the GMM, with respect to the GMM parameters and compare N descriptors to k visual words.

As compared to other methods, the FV representation offers certain advantages such as low computational requirements as it can be computed from small vocabularies and better performance with linear classifiers as they do not generate complex boundaries between the classes. This property results in significant advantage as it allows us to use linear classifiers that learn very efficiently and are fast to compute.

2.3.2. Support vector machines (SVM)

A supervised learning method, SVM works by generating a separating hyperplane [18]. The decision surface separating the classes is a hyperplane of the form:

$$\mathbf{W}^{\mathbf{T}}\mathbf{X} + b = 0 \tag{10}$$

Where $\mathbf{X}$ is the input vector, $\mathbf{W}$ is the weight vector, and b is the bias. The algorithm works by creating an optimal hyperplane using the labeled training data that is provided to it during the training

phase. This hyperplane then categorizes new test data. Thus, the SVM is a linear, non-probabilistic binary classifier. The binary SVM can be extended to work as a multi-class classifier. Two of the most common approaches for this are one-versus-one and one-versus-all. This paper employs the one-versus-all approach.

2.3.3. K-nearest neighbors (KNN)

KNN is a supervised learning algorithm that stores all the available feature vectors during the training phase and assigns labels to the images and classifies new feature vectors based on a similarity score (e.g., distance metric) [20]. In the classification phase, the unlabeled point is just given the label of the K Nearest Neighbors to the point.

$$y = \frac{1}{K}\sum_{i:1}^{K} y_i \tag{11}$$

Typically a majority vote involving K Nearest Neighbors of the new query point is used for the classification.

2.3.4. Decision trees (DT)

DTs also belong to the class of supervised ML. DT works by continuously splitting the data in accordance with a certain parameter [19]. The tree consists of two parts called the decision nodes and leaves. The leaves represent the final decision, while the nodes are points where the data is split. The decision tree used in this paper grows by the classification and regression (CART) algorithm [49]. Here, the binary DT is constructed in a top-down manner, starting with a root node obtained from any of the variables from the feature space and grows by minimizing a certain impurity metric of the two sibling nodes as per a specified splitting rule. To avoid the over-fitting problem, the splits number is constrained to be lower than the number of categories. This effectively acts as the pre-prune method. The missing values are taken care using surrogate splitting, where other input variables values are used to do a split for observations with a missing value for the best split.

2.4. Deep learning approach

DL techniques based on ANNs have been showed to be very efficient at solving many computer vision problems. The fundamental difference between traditional ML and CNN is that CNN does not require hand-crafted features. CNN's can extract features directly from raw images by tuning the parameters in the convolutional and the pooling layers. DL techniques such as the CNNs introduced by LeCun [23] have an architecture specially made for images. This has resulted in a recent rise in the use of CNNs for image processing applications. In this section, we briefly discuss CNN architectures.

2.4.1. Proposed convolutional neural network (CNN)

The CNN model employed in this paper for classifying plant stress was based on the model proposed in [50]. ConvNet has both the feature extraction and the classification capabilities. The groups of arrays that form the input and the output of each stage are refereed to as feature maps. Each feature map at the output presents a particular feature extracted for all the locations on the input. Each stage has three layers, namely, filter bank layer, non linearity layer, and feature pooling layer. A typical ConvNet can have one, two, or three such stages. These layers are followed by a classification stage. Our CNN network has 23-layered architecture as shown in Fig. 3. The architecture of the proposed model is structured as (conv8-bn-relu), max-pool, (conv16-bn-relu), max-pool, (conv32-bn-relu), max-pool,

(conv64-bn-relu), max-pool, (conv128-bn-relu), fully connected, softmax, and classification output layer. The conv(n) represents a 2d convolution layer with kernel size as $3x3$ and stride 1. These filters slide across the input image both horizontally and vertically and the dot product is computed at every spatial location known as activation map. The output of the convolution layer is obtained by stacking all the activation maps across the depth dimension. The 'bn' is the batch normalization layer. The relu (ReLU) stands for rectified linear unit. ReLU perform a nonlinear mapping onto the results of the preceding layers. The max-pooling layer has kernel size $2x2$ and stride 2. They are used to reduce the dimensionality of convolutional layers. The fully connected layers provide a global representation for the image by summarizing the feature vectors from the preceding layers. The classification layer categorizes the computed feature vectors formed at the fully connected layer into the image class using an appropriate loss function. The functionality of different layers is briefly summarized below. Mathematically '$conv\,(n)$' is defined as:

$$C\,(p,q) = (I * w)\,(p,q) = \sum_{k}\sum_{I} I\,(p-k, q-l)\,w\,(k,l). \tag{12}$$

In the above equation, I is the input image with size (p,q), w is the convolutional kernel with dimensions (k,l).

The functionality of 'bn' layer is mathematically given as:

$$\bar{x} = \frac{x_i - \mu_{mB}}{\sqrt{\sigma_{mB}^2 + \in}}, \tag{13}$$

where x_i is the input, μ_{mB} and σ_{mB} are the mean and the variance over a mini-batch and over each input channel. $\in$ is added to improvise the numerical stability when the mini-batch variance is very small.

$$y_i = \gamma \bar{x_i} + \beta. \tag{14}$$

The activations are further shifted by offset β and scaled by factor γ. Both β and γ are learnable parameters updated during network training.

ReLu counteracts for the gradient vanishing problem and is defined as:

$$\mathrm{ReLu}\,(x) = \max\,\{x, 0\}\,. \tag{15}$$

ReLu's gradient is 1 for the input not less than 0. Using ReLU for activation in CNN leads to faster convergence. Max pooling (max-pool) is given by

$$h_{i,j} = \max_{k,l}\; x_{i+k-1,j+l-1}, \quad 1 \le k, l \le m, \tag{16}$$

where m is the kernel width, and the maximization is simultaneously over k and l.

The Softmax layer constrains the output in the range $(0,1)$ and it is mathematically given as:

$$\mathrm{soft}\,(x_i) = \frac{\exp x_i}{\sum_{j=1}^{n} \exp x_j}. \tag{17}$$

2.5. Performance evaluation metrics

The performance of the proposed CNN model is evaluated using the performance metrics of Accuracy(Acc), Sensitivity(Se), Specificity(Sp) and Precision(Pre). Mathematically they are defined as;

$$\mathrm{Accuracy}\,(Acc) = \frac{\text{Total correct classifications}}{\text{Total samples}} \tag{18}$$

$$\mathrm{Sensitivity}\,(Se) = \frac{\text{Total true positives}}{\text{Total positives}} \tag{19}$$

$$\mathrm{Specificity}\,(Sp) = \frac{\text{Total true negatives}}{\text{Total negatives}} \tag{20}$$

$$\mathrm{Precision}\,(\mathrm{Pr}) = \frac{\text{Total correct classifications}}{\text{Total classifications made}} \tag{21}$$

3. Results and discussions

3.1. Results

In this section, we have evaluated and compared the classification performance of both classical ML techniques and proposed CNN. It is significant to mention that before performing the ML and DL experiments, the image size is reduced. Images are reduced to a size of

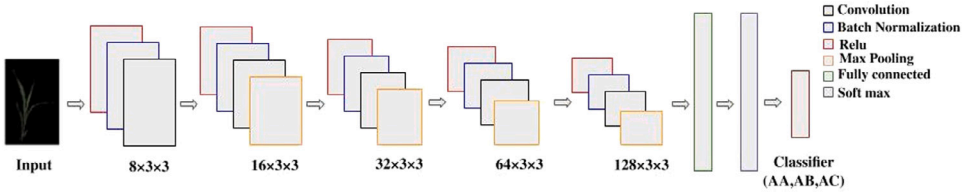


Fig. 3. CNN architecture used in this paper.

270 × 270, depth 3 (RGB channels) with resolution as 72 pixels/inch for both ML and DL experiments so as to lower the computational cost. Also, as the dataset is made up of images of different plant ages from seedling till maturity, we used a pixel count threshold to remove very short plants which are mainly 3 days old. In this stage, at first, we find the mean value of every image obtained from the segmentation stage. Later, the mean value is compared with the threshold value equal to 300. Only images having mean values above this threshold are used for our experiments. Additionally, for all the experiments 60% data is used for training and 40% for validation. We have also evaluated the performance of CNN model on plant images with background and without background. The obtained results were also compared with existing state of art deep architectures like ResNet18 [51,52] and NasNet large [53]. These proposed model is fit for 50 epochs (determined empirically). Further the weight parameters are trained using $'sgdm'$ with the initial learning rate as 0.001. Binary cross-entropy is used as a loss function. The models are implemented in MATLAB 18b platform and executed on a system with configuration as Intel Core $i7-4500U$ CPU, 3.40 GHz, and 16 GB RAM.

3.1.1. Performance evaluation of the classical ML techniques

To evaluate the performance of classical ML techniques, firstly the local descriptors were extracted from the segmented plant images, and then they were fed to the classifiers (SVM, KNN, and DT) for categorization. Figs. 4, 5, and 6 show the performance of the SVM, KNN and DT classifiers respectively for SIFT and HOG features, in terms of confusion matrices for classifying the plants categories AA, AB and AC corresponding to three nitrogen levels. The confusion matrices are represented in the form of heatmaps. The colors are based on a count aggregation, which totals the number of times each pair of 'x' and 'y' values appear together in the table. The 'x' input indicates the table variable to display along the x-axis, i.e., predicted AA, AB, and AC. The 'y' input indicates the table variable to display along the y-axis, i.e., actual AA, AB, and AC. Larger the count, deeper the color. For all the subsequent confusion matrices shown in the paper, the same convention is followed. From the confusion matrices, it is evident that all classifiers yield ceiling level performance with HOG features in both views and classes (AC and AB). However for class AA, SIFT features outperformed the HOG features. The performance metrics for the two different views and for SIFT and HOG features have been presented separately in Table 2. From the tubular results presented, it is seen that HOG features rendered the best classification performance. The comparative analysis among classifiers shows that SVM outperforms other classifiers by attaining an average classification accuracy of 74% and 75% using SIFT and HOG features. The limitation of using SVM is that the computation time is large. As shown in last column of Table 2, the computation time is 570.3 s for SVM using HOG features. It also indicates that a trade-off between performance and speed can always be considered depending on the need of application.

3.1.2. Performance of the CNN for images with background

In this section, we have used images with background information to check the robustness of the proposed CNN model. Fig. 7 represents the result of the proposed CNN method for plant images with background. As evident, the proposed method resulted in appreciable number of true positives in each class thereby justifying the robustness.

Table 2
Performance metrics for the classical Machine Learning techniques (Acc: Accuracy, Se: Sensitivity, Sp: Specificity and Pre: Precision) and the average time per s.

Classifier	Feature	V. Angle	Acc	Se	Sp	Pre	Average time (s)
SVM	SIFT	0^o	0.73	0.75	0.77	0.75	3499.1
		90^o	0.74	0.73	0.75	0.83	
	HOG	0^o	0.75	0.72	0.86	0.83	570.3
		90^o	0.74	0.75	0.87	0.82	
KNN	SIFT	0^o	0.52	0.75	0.79	0.64	282.4
		90^o	0.51	0.72	0.70	0.58	
	HOG	0^o	0.76	0.76	0.90	0.72	249.7
		90^o	0.73	0.78	0.87	0.73	
DT	SIFT	0^o	0.64	0.60	0.82	0.64	220.8
		90^o	0.66	0.59	0.77	0.60	
	HOG	0^o	0.70	0.62	0.85	0.66	198.5
		90^o	0.71	0.68	0.81	0.70	

Table 3
Performance metrics for proposed CNN, ResNet-18 and NasNet Large (Acc: Accuracy, Se: Sensitivity, Sp: Specificity and Pre: Precision).

Classifier	V. Angle	Acc	Se	Sp	Pre
Proposed CNN (With background)	0^o	0.75	0.52	0.81	0.64
	90^o	0.72	0.50	0.80	0.56
Proposed CNN (Without background)	0^o	0.83	0.66	0.89	0.76
	90^o	0.82	0.64	0.88	0.76
	Both views	0.84	0.63	0.89	0.71
ResNet18	0^o	0.84	0.64	0.88	0.76
	90^o	0.82	0.63	0.87	0.73
NasNet large	0^o	0.87	0.75	0.91	0.81
	90^o	0.84	0.68	0.92	0.70

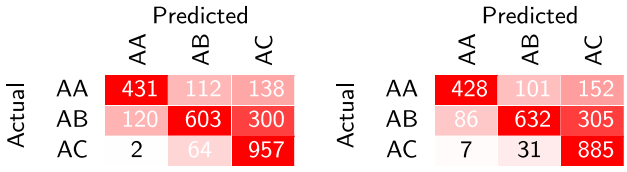
Table 4
The average time per image for SIFT and HOG features for both kind of images — with background and without background-.

Feature	Time	
	With background	Without background
SIFT	4.0866 s	1.3788 s
HOG	1.4485 s	1.0758 s

Table 5
Comparison of the parameter numbers.

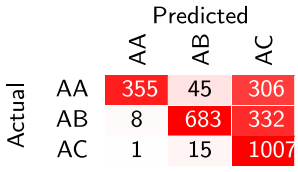
Type of CNN model	Parameter numbers	Storage requirement
NasNet large	84,912,645	343 MB
ResNet18	11,182,851	40.71 MB
Proposed CNN	197,187	1.28 MB

Table 3, shows that the classification accuracy of CNN for plant images with background for both 0° and 90° views are 75% and 72% respectively. Also, the proposed CNN achieves a sensitivity, specificity, and precision equal to 52%, 81%, 64% for 0° view and 50%, 80%, 56% for 90° view. Higher values of specificity and lower value of sensitivity indicate that the proposed CNN has resulted in fewer false positives in contrast to false negatives for the images with background (see Fig. 12).

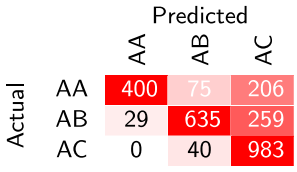


(a) SIFT: 0^o view

(b) SIFT: 90^o view

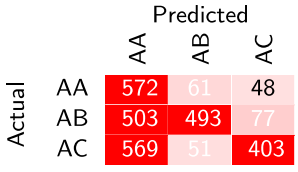


(c) HOG: 0^o view

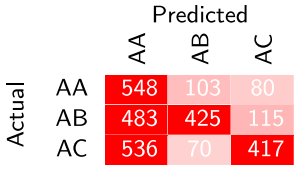


(d) HOG: 90^o view

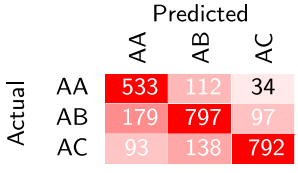
Fig. 4. Confusion matrices depicting the results for the SVM classifier for SIFT and HOG descriptors. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



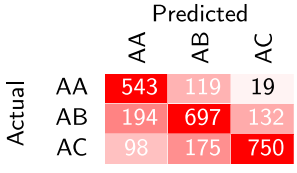
(a) SIFT: 0^o view



(b) SIFT: 90^o view

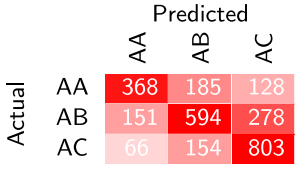


(c) HOG: 0^o view

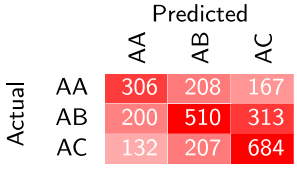


(d) HOG: 90^o view

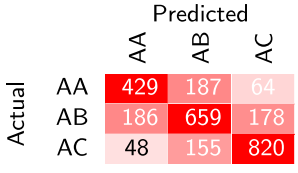
Fig. 5. Confusion matrices depicting the results for the KNN classifier for SIFT and HOG descriptors. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



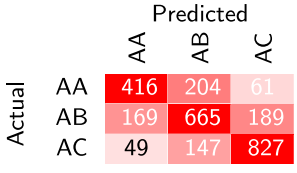
(a) SIFT: 0^o view



(b) SIFT: 90^o view

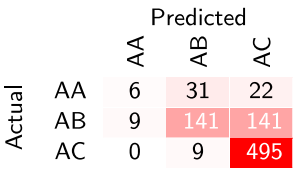


(c) HOG: 0^o view

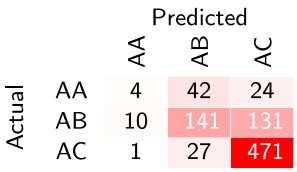


(d) HOG: 90^o view

Fig. 6. Confusion matrices depicting the results for the DT classifier for SIFT and HOG descriptors. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

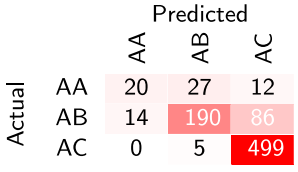


(a) CNN: 0^o view

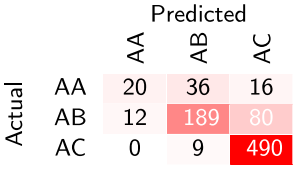


(b) CNN: 90^o view

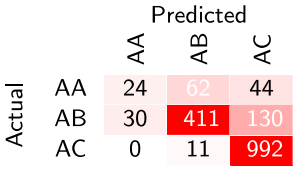
Fig. 7. Confusion matrices depicting the results of the plant images with background for the proposed CNN for both 0° and 90° angles.



(a) CNN: 0^o view

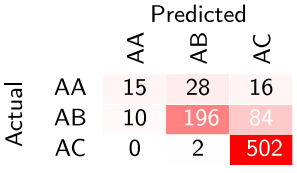


(b) CNN: 90^o view

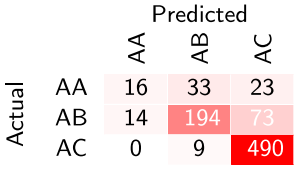


(c) CNN:Both views

Fig. 8. Confusion matrices depicting the results of the plant images without background for the proposed CNN for 0°, 90° angles, and both views combined (0° and 90° angles)

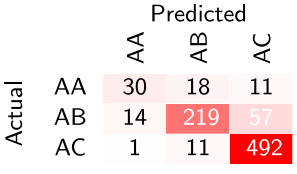


(a) CNN: 0^o view

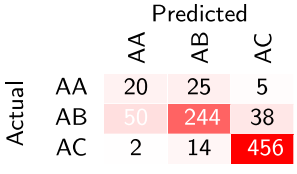


(b) CNN: 90^o view

Fig. 9. Confusion matrices depicting result for plant images without background using ResNet18 for both 0° and 90° angles.



(a) CNN: 0^o view



(b) CNN: 90^o view

Fig. 10. Confusion matrices depicting result for NasNet large model for plant images without background in 0° and 90° angles.

3.1.3. Performance of the CNN on images without background

Fig. 8 represents the result of the CNN method for plant images without a background. The classification accuracy, sensitivity, specificity, and precision for plant images without background are shown in Table 3. The classification accuracy's of proposed CNN for 0° view, for 90° view and for both 0° and 90° views are 83%, 82% and 84% respectively. Moreover, other performance evaluation metrics have a value equal to 66%, 89%, 76% for 0° view, 64%, 88%, 76% for 90° view, and 63%, 89%, 71% for both the views. There is an increase of approx. 10% in contrast to results with background for both 0° and 90°

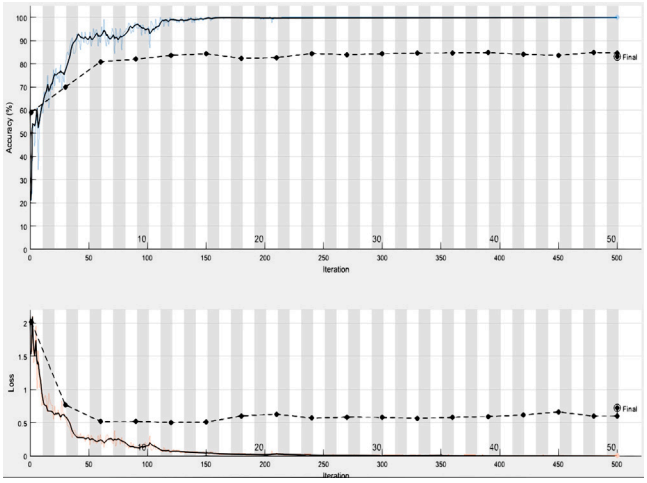


Fig. 11. Accuracy and loss versus the number of iterations (0° view).

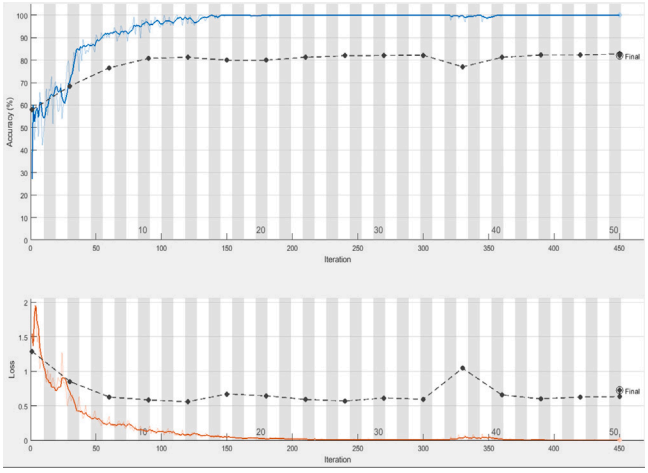


Fig. 12. Accuracy and loss versus the number of iterations (90° view).

view, which is justifying the advantage of the removing background. The possible reason for the increase in accuracy after background subtraction is that the size of the plant is very small as compared to that of background. Background subtraction enables the descriptors to be extracted only from the plant itself rather than the plant as well as the invariant background, thereby facilitating fast and accurate classification with larger true positives and fewer misclassifications. We noticed that in combining both 0° and 90° views, there is an increase in accuracy by 1% and 2% in comparison to 0° and 90° view. Moreover, for the specificity there is an increase of 1% on combining both the views in contrast to the 90° view. One of the probable reasons for such an increase is that more number of images are given as input to the proposed CNN model. Classification accuracy and loss versus the number of iterations for 0° and 90° were illustrated in Fig. 11 and in Fig. 12. It is clearly evident that there is an improvement in performance with an increase in the number of iterations. This improvement is rapid initially till 10 epochs followed by a plateaus showing minor improvement in validation accuracy. The training progress curves shown in Fig. 11 indicates a scope of performance improvement in the proposed CNN model by adding an early stopping criterion (like validation patience), image augmentation, and L2 Regularization to account for over-fitting.

3.1.4. Performance comparison of the proposed CNN model with established deep architectures

This subsection illustrates the comparison of the proposed 23-layered CNN model with the well established deep models like ResNet18 [51] and NasNet Large [53]. The confusion matrices are shown in Figs. 9 and 10 and the performance metrics are reported in Table 3. As seen from Table 3, applying ResNet18 and NasNet large to images without background resulted in an accuracy of 83.59%, 86.87% for 0° and 82.16%, 84.27% for 90° view. Other metrics for ResNet18 have a value of 64%, 88%, 76% for 0° and 63%, 87%, 73% for 90° view. For NasNet large, the metrics have surged to 75%, 91%, 81% for 0° and 68%, 92%, 70% for 90° view. The results obtained via ResNet18 for both the views are comparable to those achieved using the proposed deep network. In contrast to the NasNet large model having 1234 layers with millions of learnable parameters, the proposed 23-layered CNN model achieved slightly less accuracy. Accuracy using NasNet large is better by nearly 4.55% and 2.76% in contrast to our proposed architecture that has just 23 layers and substantially less learnable parameters. The results signify that a 23-layered model is capable of measuring stress in plants with competitive performance to that of a 71-layered and 1234-layered model. Additionally, it is observed that employing a 71-layered and 1234 layered model surges the computational cost by a factor of 4.51 and 22.18 in contrast to the proposed 23-layered model. A comparison of the learnable parameter numbers for proposed and the deep architectures taken for comparison is given in Table 5. From, the tabular findings, it is evident that the proposed CNN requires a small number of parameters than the other established deep architectures. The number of learnable parameters in the proposed model is just 1,97,187 in contrast to 11 million and 84 million learnable parameters for ResNet18 and NasNet large architectures. Clearly, due to the small number of learnable parameters, the computational and disk storage requirement to train and store the proposed model is substantially lower than other investigated architectures. As indicated in Table 5, it is just 1.2 MB in contrast to 40.71 MB and 343 MB for the other deep architectures taken for comparison. Deeper architectures not only require high disk space but also need high-performance GPU/CPUs for training. Lesser number of learnable parameters and smaller disk storage requirements make the proposed 23-layered model practical for the real-time applications as it can be easily implementable on a simple computing platform.

3.2. Discussion

The results obtained in our experimental studies prove that the proposed CNN outperformed other considered traditional ML methods. Based on all the above results, we found that the proposed CNN can learn the features automatically for efficient classification of stress levels in plants. Hence, the proposed CNN found to be the best tool in assisting researchers for precise plant stress classification. As evident from other studies, Deeper layers improve the performance, but require huge amounts of plant datasets that are usually not available publicly. The dataset we used in this paper is the only publicly available dataset containing plant shoot images under nitrogen deficiency. Due to the lack of required data, we are in the process of preparing our own plant shoot images dataset for quantifying and classifying different abiotic stress levels using Deep Learning methods in our future work. There are few related studies in the existing literature that have dealt with plant stress detection. In [21], Zhuang et al. have considered the effect of water stress conditions for maize plant on fields using ML techniques. In [41], Jiangyong et al. have used the same maize dataset as [21], to identify and classify drought stress by using DCNN. However, the dataset used in their work is not publicly available. The only reference in the existing literature that has used the same dataset as ours is work by Veley et al. [45]. However, the authors have used simple plant profiling techniques like change in height, width and color fluctuations to report growth rate with respect to Nitrogen deficiency. Therefore, a

direct comparison with their work is not possible. Also, the proposed CNN model attains a comparable performance to the established deep architectures. Prominently, in comparison to NasNet large the proposed model has simple architecture, less number of trainable parameters, requires less storage space, and can be easily implemented on a personnel computer making it practical for the real time scenario. On the other hand, NasNet large has 1234 layers, millions of trainable parameters, requires more disk space for storage and necessitate the requirement of powerful CPU/GPU for its implementation. All these make NasNet large computationally very complex for the application of automatic image based plant phenotyping. Also, it is well known that the SIFT and HOG features and even the DL methods are usually capable of ignoring the invariant features that correspond to the background. However, detecting the keypoints that belong to the image background and extracting the features for those keypoints results in a significant increase in the computation required and the time consumed. This makes the plant segmentation a crucial step in enhancing the performance of both the classical ML as well as the DL methods for nitrogen stress classification. Subtracting the background requires very less computation but results in a significant reduction in the overall computation required for the image feature extraction. This is evident from the time required for extracting features, as seen in Table 4, which is reduced to just 1 s for HOG features. As we have a huge number of images, and size of features is quite big, with background subtraction, we are able to significantly reduce the number of keypoints and consecutively, the computational load as well as computation time.

4. Conclusions

In this present work, we have demonstrated the use of DL in classification of stress level in Sorghum plant from the shoot images. The performance of the proposed 23-layered CNN is compared with other established ML techniques and deep architectures like ResNet18 and NasNet Large. Here, we observed that the proposed 23-layered CNN model outperformed the conventional ML techniques in identifying plant stress levels. Our experimental results demonstrate that the proposed CNN is capable of providing better feature representations for nitrogen induced stress classification as compared to local features used in the classical ML methods. Also, the proposed model achieves comparable performance over well-established deep architectures with much less trainable parameters. In our future work, we will extend this CNN based model to other plant species with both offline and online images.

CRediT authorship contribution statement

Shiva Azimi: Experiment designing, Conceptualization, Methodology, Creation of models and analysis, Writing - original draft. **Taranjit Kaur:** Methodology, Creation of models and analysis, Writing - original draft. **Tapan K. Gandhi:** Experiment designing, Conceptualization, Methodology, Creation of models and analysis, Writing - original draft.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] N. Li, H. Lin, T. Wang, Y. Li, Y. Liu, X. Chen, X. Hu, Impact of climate change on cotton growth and yields in xinjiang, China, Field Crops Res. (2019) 107590.

[2] F. Tardieu, L. Cabrera-Bosquet, T. Pridmore, M. Bennett, Plant phenomics, from sensors to knowledge, Curr. Biol. 27 (15) (2017) R770–R783.

[3] C. Zhou, J. Le, D. Hua, T. He, J. Mao, Imaging analysis of chlorophyll fluorescence induction for monitoring plant water and nitrogen treatments, Measurement 136 (2019) 478–486.

[4] K. Neumann, C. Klukas, S. Friedel, P. Rischbeck, D. Chen, A. Entzian, N. Stein, A. Graner, B. Kilian, Dissecting spatiotemporal biomass accumulation in barley under different water regimes using high-throughput image analysis, Plant Cell Environ. 38 (10) (2015) 1980–1996.

[5] M.S.M. Asaari, P. Mishra, S. Mertens, S. Dhondt, D. Inzé, N. Wuyts, P. Scheunders, Close-range hyperspectral image analysis for the early detection of stress responses in individual plants in a high-throughput phenotyping platform, ISPRS J. Photogramm. Remote Sens. 138 (2018) 121–138.

[6] M. Minervini, H. Scharr, S.A. Tsaftaris, Image analysis: the new bottleneck in plant phenotyping [applications corner], IEEE Signal Process. Mag. 32 (4) (2015) 126–131.

[7] G. Bai, Y. Ge, W. Hussain, P.S. Baenziger, G. Graef, A multi-sensor system for high throughput field phenotyping in soybean and wheat breeding, Comput. Electron. Agric. 128 (2016) 181–192.

[8] R. Panwar, K. Goyal, N. Pandey, N. Khanna, Imaging system for classification of local flora of Uttarakhand region, in: 2014 International Conference on Power, Control and Embedded Systems, ICPCES, IEEE, 2014, pp. 1–6.

[9] J.F. Humplík, D. Lazár, A. Husičková, L. Spíchal, Automated phenotyping of plant shoots using imaging methods for analysis of plant stress responses–a review, Plant Methods 11 (1) (2015) 29.

[10] E. Hamuda, M. Glavin, E. Jones, A survey of image processing techniques for plant extraction and segmentation in the field, Comput. Electron. Agric. 125 (2016) 184–199.

[11] A. Singh, B. Ganapathysubramanian, A.K. Singh, S. Sarkar, Machine learning for high-throughput stress phenotyping in plants, Trends Plant Sci. 21 (2) (2016) 110–124.

[12] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436.

[13] F.M. Kheirkhah, H. Asghari, Plant leaf classification using GIST texture features, IET Comput. Vis. 13 (4) (2018) 369–375.

[14] X. Qiao, J. Bao, H. Zhang, F. Wan, D. Li, FvUnderwater sea cucumber identification based on Principal Component Analysis and Support Vector Machine, Measurement 133 (2019) 444–455.

[15] G. Dhingra, V. Kumar, H.D. Joshi, A novel computer vision based neutrosophic approach for leaf disease identification and classification, Measurement 135 (2019) 782–794.

[16] Y. Zheng, Q. Zhu, M. Huang, Y. Guo, J. Qin, Maize and weed classification using color indices with support vector data description in outdoor fields, Comput. Electron. Agric. 141 (2017) 215–222.

[17] G. Saleem, M. Akhtar, N. Ahmed, W. Qureshi, Automated analysis of visual leaf shape features for plant classification, Comput. Electron. Agric. 157 (2019) 270–280.

[18] C.J. Burges, A tutorial on support vector machines for pattern recognition, Data Min. Knowl. Discov. 2 (2) (1998) 121–167.

[19] S.R. Safavian, D. Landgrebe, A survey of decision tree classifier methodology, IEEE Trans. Syst. Man Cybern. 21 (3) (1991) 660–674.

[20] T.M. Cover, P.E. Hart, et al., Nearest neighbor pattern classification, IEEE Trans. Inform. Theory 13 (1) (1967) 21–27.

[21] S. Zhuang, P. Wang, B. Jiang, M. Li, Z. Gong, Early detection of water stress in maize based on digital images, Comput. Electron. Agric. 140 (2017) 461–468.

[22] H.S. Naik, J. Zhang, A. Lofquist, T. Assefa, S. Sarkar, D. Ackerman, A. Singh, A.K. Singh, B. Ganapathysubramanian, A real-time phenotyping framework using machine learning for plant stress severity rating in soybean, Plant Methods 13 (1) (2017) 23.

[23] Y. LeCun, B.E. Boser, J.S. Denker, D. Henderson, R.E. Howard, W.E. Hubbard, L.D. Jackel, Handwritten digit recognition with a back-propagation network, in: Advances in Neural Information Processing Systems, 1990, pp. 396–404.

[24] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, et al., Gradient-based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324.

[25] G. Litjens, T. Kooi, B.E. Bejnordi, A.A.A. Setio, F. Ciompi, M. Ghafoorian, J.A. Van Der Laak, B. Van Ginneken, C.I. Sánchez, A survey on deep learning in medical image analysis, Med. Image Anal. 42 (2017) 60–88.

[26] A. Kamilaris, F.X. Prenafeta-Boldú, Deep learning in agriculture: A survey, Comput. Electron. Agric. 147 (2018) 70–90.

[27] Y.-H. Wu, L. Shang, Z.-K. Huang, G. Wang, X.-P. Zhang, Convolutional neural network application on leaf classification, in: International Conference on Intelligent Computing, Springer, 2016, pp. 12–17.

[28] S.H. Lee, C.S. Chan, S.J. Mayo, P. Remagnino, How deep learning extracts and learns leaf features for plant classification, Pattern Recognit. 71 (2017) 1–13.

[29] S.H. Lee, C.S. Chan, P. Remagnino, Multi-organ plant classification based on convolutional and recurrent neural networks, IEEE Trans. Image Process. 27 (9) (2018) 4287–4301.

[30] L.C. Uzal, G.L. Grinblat, R. Namías, M.G. Larese, J. Bianchi, E. Morandi, P.M. Granitto, Seed-per-pod estimation for plant breeding using deep learning, Comput. Electron. Agric. 150 (2018) 196–204.

[31] S. Madec, X. Jin, H. Lu, B. De Solan, S. Liu, F. Duyme, E. Heritier, F. Baret, Ear density estimation from high resolution RGB imagery using deep learning technique, Agricult. Forest Meteorol. 264 (2019) 225–234.

[32] Y. Sun, Y. Liu, G. Wang, H. Zhang, Deep learning for plant identification in natural environment, Comput. Intell. Neurosci. 2017 (2017).

[33] J.G.A. Barbedo, Plant disease identification from individual lesions and spots using deep learning, Biosyst. Eng. 180 (2019) 96–107.

[34] H. Durmuş, E.O. Güneş, M. Kırcı, Disease detection on the leaves of the tomato plants by using deep learning, in: 2017 6th International Conference on Agro-Geoinformatics, IEEE, 2017, pp. 1–5.
[35] X. Zhang, Y. Qiao, F. Meng, C. Fan, M. Zhang, Identification of maize leaf diseases using improved deep convolutional neural networks, IEEE Access 6 (2018) 30370–30377.
[36] E. Suryawati, R. Sustika, R.S. Yuwana, A. Subekti, H.F. Pardede, Deep structured convolutional neural network for tomato diseases detection, in: 2018 International Conference on Advanced Computer Science and Information Systems, ICACSIS, IEEE, 2018, pp. 385–390.
[37] S. Ghosal, D. Blystone, A.K. Singh, B. Ganapathysubramanian, A. Singh, S. Sarkar, An explainable deep machine vision framework for plant stress phenotyping, Proc. Natl. Acad. Sci. 115 (18) (2018) 4613–4618.
[38] K. Kuwata, R. Shibasaki, Estimating crop yields with deep learning and remotely sensed data, in: 2015 IEEE International Geoscience and Remote Sensing Symposium, IGARSS, IEEE, 2015, pp. 858–861.
[39] L. Song, S. Prince, B. Valliyodan, T. Joshi, J.V.M. dos Santos, J. Wang, L. Lin, J. Wan, Y. Wang, D. Xu, et al., Genome-wide transcriptome analysis of soybean primary root under varying water-deficit conditions, BMC Genom. 17 (1) (2016) 57.
[40] G. Sehgal, B. Gupta, K. Paneri, K. Singh, G. Sharma, G. Shroff, Crop planning using stochastic visual optimization, in: 2017 IEEE Visualization in Data Science, VDS, IEEE, 2017, pp. 47–51.
[41] J. An, W. Li, M. Li, S. Cui, H. Yue, Identification and classification of maize drought stress using deep convolutional neural network, Symmetry 11 (2) (2019) 256.
[42] D.G. Lowe, et al., Object recognition from local scale-invariant features, in: Iccv, Vol. 99, No. 2, 1999, pp. 1150–1157.
[43] S. Azimi, B. Lall, T.K. Gandhi, Performance evalution of 3D keypoint detectors and descriptors for plants health classification, in: 2019 16th International Conference on Machine Vision Applications, MVA, IEEE, 2019, pp. 1–6.
[44] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: International Conference on Computer Vision & Pattern Recognition, Vol. 1, CVPR'05, IEEE Computer Society, 2005, pp. 886–893.
[45] K.M. Veley, J.C. Berry, S.J. Fentress, D.P. Schachtman, I. Baxter, R. Bart, High-throughput profiling and analysis of plant responses over time to abiotic stress, Plant Direct 1 (4) (2017) e00023, https://plantcv.danforthcenter.org/pages/datasets/sorghum_abiotic_stress.html.
[46] J.A. Hartigan, M.A. Wong, Algorithm AS 136: A k-means clustering algorithm, J. R. Stat. Soc. Ser. C Appl. Stat. 28 (1) (1979) 100–108.
[47] C. Kim, Cook's distance in spline smoothing, Stat. Probab. Lett. 31 (2) (1996) 139–144.
[48] F. Perronnin, C. Dance, Fisher Kernels on visual vocabularies for image categorization, in: 2007 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2007, pp. 1–8.
[49] L. Breiman, Classification and Regression Trees, Routledge, 2017.
[50] M. Rahnemoonfar, C. Sheppard, Deep count: fruit counting based on deep simulated learning, Sensors 17 (4) (2017) 905.
[51] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition. Computer Vision and Pattern Recognition (CVPR), in: 2016 IEEE Conference on, Vol. 5, 2015, p. 6.
[52] O.A. Penatti, K. Nogueira, J.A. Dos Santos, Do deep features generalize from everyday objects to remote sensing and aerial scenes domains?, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2015, pp. 44–51.
[53] B. Zoph, V. Vasudevan, J. Shlens, Q.V. Le, Learning transferable architectures for scalable image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 8697–8710.