Artificial intelligence-powered expert system model for identifying fall armyworm infestation in maize (Zea mays L.)

R. Prabha
Department of Agricultural Entomology, Agricultural College and Research Institute, Tamil Nadu Agricultural University, Coimbatore - 641003 (Tamil Nadu), India

J. S. Kennedy*
Department of Agricultural Entomology, Agricultural College and Research Institute, Tamil Nadu Agricultural University, Coimbatore - 641003 (Tamil Nadu), India

G. Vanitha
Department of Physical Science and Information Technology, Agricultural College and Research Institute, Tamil Nadu Agricultural University, Coimbatore - 641003 (Tamil Nadu), India

N. Sathiah
Department of Agricultural Entomology, Agricultural College and Research Institute, Tamil Nadu Agricultural University, Coimbatore - 641003 (Tamil Nadu) India

M. Banu Priya
Department of Agricultural and Information Technology, Agricultural College and Research Institute, Tamil Nadu Agricultural University, Coimbatore - 641003 (Tamil Nadu) India

*Corresponding author. Email: jskennedy@tnau.ac.in

How to Cite

Abstract
Maize (Zea mays L.) is one of the most saleable cereal crops grown worldwide and a dominant staple food in many developing countries. The severe outbreak of fall armyworm in maize causes massive yield loss. Modern technologies, including smartphones, can assist in detecting recognising the fall armyworm infestation in maize. The objective of this study was to develop an automated Artificial Intelligence Powered Expert System (AIPES) for identifying fall armyworm infestation in maize. In addition, it put forward a deep learning-based model that is trained on photographs of healthy and fall armyworm infested leaves, cobs and tassels from a dataset and furnished an application that will be detecting maize fall armyworm infestation using Convolutional Neural Network (CNN) architecture and Mobile Net V2 framework model. The study developed an Artificial Intelligence (AI) based maize fall armyworm infestation detection system using a DCNN (Deep Convolutional Neural Network) to support maize cultivating farmers. The model executed the objective by accurately identifying the fall armyworm infested maize plant and also classified them vis-c-vis the healthier crop. The deep learning models were trained to detect and recognise fall armyworm infection using more than 11000 images of fall armyworm infected leaves, cobs, and tassels. The created application (AIPES for identifying fall armyworm infestation in maize) using CNN detected and recognised the fall armyworm infestation in maize with a 100 per cent training accuracy rate and 87 per cent validation accuracy. So, the detection of maize fall armyworm and the treatment of fall armyworm-infested maize could lead to a higher maize crop yield.

Keywords: Artificial intelligence, Convolutional neural networks, Deep learning, Fall armyworm infestation detection, Maize, Mobile app

INTRODUCTION
Maize has emerged as a crucial cereal crop in peninsular India, where states like Andhra Pradesh, which ranks fifth in terms of area (0.79 m ha), have recorded the highest production (4.14 m t) and productivity (5.26 t ha\(^{-1}\)) in the country, despite the fact that productivity in some of Andhra Pradesh's districts is more or equal to that of the United States (Murdia et al., 2016). In the recent past, maize crop has been imperilled by the fall armyworm (Spodoptera frugiperda, J.E. Smith) (FAW) that has caused substantial yield loss in maize. Fall armyworm is native to tropical and subtropical America and is acknowledged as a sporadic pest in the United States since 1797. The spread of fall FAW in India has been proclaimed since it is documented in Karnataka in
MATERIALS AND METHODS

Architecture of image processing system

The present system of Image processing system comprised of five phases namely as image acquisition, image pre-processing, data augmentation, feature extraction and classification as used by Akhtar et al. (2013).

Image acquisition

A large dataset was used to classify the maize fall armyworm infested leaves, cobs, tassels. Maize FAW infested leaves, cobs and tassels dataset of JPEG images was shot with a Nikon D7500 P-Digital Camera. A total of more than 11000 photographs with varying severity levels of infestations were shot from FAW infested maize field and stored in the system for image processing by following the methodology described (Kulkarni, 2018; Militante, 2019; Syarief and Setiawan, 2020). Dataset was collected from various maize growing research plots of TNAU and various blocks (Perunthuraipattu, Thenmudiyanur, Agarampalipattu, Allapanur, Rayandapuram, Kankayanur, Varakur and Peruman) of Tiruvannamalai districts of Tamil Nadu. Then the collected images were annotated based on the symptoms caused by the FAW (Table 1).

Image classification used for model development

Maize tassel was also classified as “Healthy tassel” and “Fall armyworm infested tassel” (Fig. k-l). The image dataset in maize is shown in Fig. 1. A leaf was identified and deemed as a “Healthy maize leaf” when it remained entirely green (Fig.1a). However, if there were pinholes on the leaves, that could be considered as the “Pinhole symptom” caused by fall armyworm (Fig.1b). “Circular hole symptom caused by the FAW” can be identified if there were any circular holes on the leaves (Fig.1c). If there were elongated lesions on the leaves, it could be considered as the “Elongated lesion-symptom” caused by fall armyworm (Fig.1d). If the whorl leaf had any fall armyworm infestation, it was considered as “Whorl leaf eaten by FAW-symptom” (Fig. 1e). Maize cob was classified based on the per cent infestation of fall armyworm (Fig.1f-j).
Table 1. Description and total number of image dataset

<table>
<thead>
<tr>
<th>Classes</th>
<th>Maize Leaves</th>
<th>Number of photos taken</th>
</tr>
</thead>
<tbody>
<tr>
<td>Healthy leaves</td>
<td></td>
<td>1000</td>
</tr>
<tr>
<td>Pinhole caused by FAW</td>
<td></td>
<td>1963</td>
</tr>
<tr>
<td>Circular hole caused by FAW</td>
<td></td>
<td>2271</td>
</tr>
<tr>
<td>Small to several lesions caused by FAW</td>
<td></td>
<td>3374</td>
</tr>
<tr>
<td>Whorl leaf is eaten by FAW</td>
<td></td>
<td>1000</td>
</tr>
<tr>
<td>Maize cob</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Classes</td>
<td>Number of photos taken</td>
<td></td>
</tr>
<tr>
<td>Nil damage to slight damage at tips of the cobs</td>
<td>196</td>
<td></td>
</tr>
<tr>
<td>&lt; 25 % of cob area showing FAW infestation</td>
<td>384</td>
<td></td>
</tr>
<tr>
<td>26 - 50 % of cob area showing FAW infestation</td>
<td>151</td>
<td></td>
</tr>
<tr>
<td>51 – 75% of cob area showing FAW infestation</td>
<td>124</td>
<td></td>
</tr>
<tr>
<td>&gt; 75 % of cob area showing FAW infestation</td>
<td>238</td>
<td></td>
</tr>
<tr>
<td>Maize tassel</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Classes</td>
<td>Number of photos taken</td>
<td></td>
</tr>
<tr>
<td>Healthy tassel</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>FAW infested tassel</td>
<td>459</td>
<td></td>
</tr>
</tbody>
</table>

Fig.1 a-l. Showing Images of the dataset in maize
**Image pre-processing**
Images were reduced in size and cropped to fit a certain input. It improves and processes the image to the required colour scale. Image pre-processing is done for reducing noise and segmenting the image, which boosts the accuracy of the CNN model. The acquired images are usually messy and un-normalized since it comes from different sources. To feed them to the algorithm, they need to be pre-processed, normalized and cleaned up. More often, image pre-processing and normalization is done to cut down the complexity and strengthen the accuracy of the applied algorithm by generating accurate results.

**Data augmentation**
When training a deep learning model, data augmentation is necessary to avoid the overfit problem produced by a small dataset. Supplemented the dataset acquired in the image acquisition to increase the dataset size and offer some variation (Random rotation, random flip, random brightness, random zoom) in image distortion. Augmentor, a Python image augmentation package for machine learning, is used in the data augmentation process.

**Feature extraction and image analysis**
Image analysis entails breaking down an image into its constituent parts to retrieve useful data. Image analysis includes finding shapes, recognising edges, eliminating noise, counting objects, and computing texture analysis or image quality. Feature extraction is a step in the feature dimensionality reduction process, which divides and reduces a large amount of raw data into smaller groupings to make processing easier. Convolution and pooling layers of Convolutional Neural Networks (CNN) are used to accomplish this process.

**Image classification**
Fully connected layers of Convolutional neural network (CNN) are used for the classification of image datasets, whereas convolutional and pooling layers are used for feature extraction. The categorization technique classifies plant portions infested with fall armyworm.

**Background knowledge**

**Convolutional Neural Network (CNN)**
The CNN is a sort of artificial neural network that is used for image identification (the capacity to recognise objects) and processing pixel data for a specific design. It consists of an input layer, many hidden layers, and an output layer in the CNN architecture. In the hidden layer, it has a convolutional layer, a Rectified Linear Unit (ReLu), a pooling layer, and a fully connected layer. A nominal process of a convolutional neural network can easily encounter and categorize the images. Through its multi-layered structure, it is compelling at evaluating graphical images and extracting the relevant features. The architecture of the CNN model was depicted in Fig.2.

**Convoision layer**
The feature map of the input image is extracted in convolution layer through a series of mathematical algorithms. The convolution layer is used to reduce the size of a 6x6 image input into a 4x4 output image matrix using 3x3 filter. ‘Convolved feature’ refers to the output matrix obtained by sliding the filter over the image from the upper left corner of the input image and computing the dot product. The results are the added values for each step multiplied by the filter’s values. The input image is converted into a new matrix with a smaller size (4x4). Fig. 3 depicts the convolution operation.
Padding
Padding is used to preserve the original dimensions of the input. In this step, zeros are added to outside of the input image. Numbers of zero layers depend upon the size of the kernel. Padding operation was depicted in Fig.4.

Pooling layers
For the down sampling layer, this layer reduces overfitting and shrinks the neuron size. Pooling operation in action was depicted in Fig.5. This layer minimises the size of the feature map, as well as the number of parameters, training time, computation rate, and overfitting. A model is said to overfit if it achieves 100% accuracy on the training dataset and 50% accuracy on the test data. To reduce the size of the feature map, ReLU and max-pooling were used.

Activation layer
Every convolution layer makes use of a non-linear ReLU (Rectified Linear Unit) activation layer. Dropout layers are also used in this layer to prevent overfitting.

Fully connected layers
This layer employs the SoftMax classifier, a well-known input classifier, to recognise and categorise maize fall armyworm infestation.

Train, validation and test dataset
In this study, the dataset of fall armyworm infested maize was split into 70 per cent, 20 per cent and 10 per cent for training, validation and testing, respectively.

Software and hardware system
The list of hardware and software used in this study are depicted in Table 2. In the Google Colaboratory editing platform, python 3 was exerted for algorithm implementation and data wrangling scripts. TensorFlow, a sophisticated framework built by Google, was practised to train and infer models. This library supported both CPU and GPU training and inference.

Google Colaboratory
Google Colaboratory is a free cloud service granted by google to anybody with a Gmail account and allows anyone to evolve and execute Python code in their browser. People who don’t have enough resources can use Google Colab to acquire a GPU for research. The
Google Colab service delivers 12.72 GB of RAM and 358.27 GB of hard disc space in one playtime. Every runtime lasts for 12 hours before being reset and the user has to re-establish a connection. This is to hinder anyone from using the GPU service to mine cryptocurrency. A new notebook was built where all Python coding was used to train the model (Fig. 6). The selection box was positioned in Runtime -> Change runtime type and appears as shown below (Fig. 7). The file was connected to a runtime. This connect button was at the top right corner of the page, as shown in the illustration below (Fig. 8).

To run a cell, Ctrl + Enter or click on the cell was used. The play button on the far left of each cell, as indicated in fig.9. The code snippet below (Fig.10) was used to link Google Colab to Google drive. The user was bestowed with a URL and a text box when the code snippet was run. The user clicked on the URL and provided Google Colab permission to access our Google Drive and its data. Once permission was granted, the user was given a one-time token that must be input into the textbox below the URL. Google Drive is contentedly mounted onto Google Colab if the token is detected and authenticated. Once

Table 2. Hardware and software used in this study

<table>
<thead>
<tr>
<th>Hardware and Software</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory</td>
<td>8.00 GB (5.88 GB usable)</td>
</tr>
<tr>
<td>Processor</td>
<td>AMD Ryzen 5, 3500U with Radeon Vega Gfx 2.10 GHz</td>
</tr>
<tr>
<td>Operating system</td>
<td>Windows 10</td>
</tr>
<tr>
<td>Editor platform</td>
<td>Google colaboratory</td>
</tr>
<tr>
<td>Deep learning libraries</td>
<td>Keras and TensorFlow</td>
</tr>
<tr>
<td>Deep learning architecture</td>
<td>Convolutional neural network</td>
</tr>
<tr>
<td>Deep learning model</td>
<td>Mobile Net V2</td>
</tr>
</tbody>
</table>
Google Drive is mounted onto Google Colab, working code into the cell could be written.

**Working with libraries**

TensorFlow is an open-source machine learning library for research that is fast, versatile and scalable. TensorFlow Lite and TensorFlow Serving, which offers the same features for mobile platforms and high-performance servers that allow to construct and train ML models on computers and mobile devices and servers. Keras is a popular open-source python neural network and it was hastily integrated into TensorFlow’s core library, allowing it to be used on top of the framework. Neural layers such as activation and cost functions, goals, batch normalisation, dropout, and pooling are a few of Keras’s building blocks and techniques for constructing a neural network.

**Steps involved in training an image dataset with Google Colaboratory**

**Uploading image dataset**

A Gmail account is required to submit image datasets and uploaded all of the image datasets to Google Drive in their original image sizes.

**Splitting image dataset in google drive**

The image datasets were divided into train and validation datasets once the uploading is completed. Two folders entitled train and validation dataset were established in the maize dataset folder.

**Import libraries in google colaboratory**

After the uploading process is complete, create a Google collaborative workspace and write Python code to train the model A few key Python libraries were imported to get started.

**Creating base dataset**

Set the base dataset and add extra layers to the model after installing the Python libraries.

**Generating model summary**

The model summary was then generated, which includes numerous layers, parameters, and gradients based on the given base dataset.

**Print the layers**

The layers were printed once the model summary has been generated.

**Mounting google drive**

Using the Google authorisation code, there is a need to mount Google Drive. So, all of the files were uploaded using this method to train the deep learning model.

**Creating training and validation dataset directory**

For both training and validation, provide a directory and set goal size, colour mode, bath size, class mode, and subset.

**Compile the model**

The model was constructed by setting the number of epochs, decay rate, learning rate, and Adam optimizer after generating the train and validation directories.

**Call back set for training the model**

A callback is a function that is passed as an argument to another function. From Keras, import the model check point and tensor board.call back.

**Training the model**

Training the dataset was done after configuring the call back function. On 10 and 100 epochs, the model was trained. As a result, training the model will take a long
time. Attention was paid to this until the end of the epoch.

**Visualize and saving the model**
Plot history can also be used to visualise the model’s output. The trained model was then saved to Google Drive as model1.hdf5, and the model inference was also shown. Then first pre-process the image dataset by resizing it, then generating the class indices.

**Prediction the dataset**
The `Imread` function must be used to upload any test dataset. After having given the cell, execute it. It will deliver the desired result.

**Testing**
A test was done to test the accuracy of the results of the classification of training data features with 100 maize FAW infestation data on each class in the research on corn plant FAW detection using the CNN approach, as shown in Table 3.

**RESULTS AND DISCUSSION**

**Output of the model**
An algorithm was trained and generated the results at 10 epochs and 100 epochs shown in (Fig.11) and (Fig.12) by using photos of symptoms of fall armyworm infestation in maize as the training dataset. It manifests the training outcome for every 10 epochs and 100 epochs.

The training data set was able to achieve 100 per cent accuracy, whereas the validation accuracy was about 74 per cent on completion of 10 epochs during training the dataset. The visualisation of train and validation accuracy plots demonstrates that the model effectively identifies and distinguishes the symptoms caused by fall armyworm in maize (Fig.11a). Train and validation loss per epoch decreased as the number of epochs increased (Fig.11b).

The model was additionally trained across 100 epochs, ensuing in a training data set that was 100 per cent accurate and a validation data set that was roughly 87 per cent accurate (Fig.12). This implies that the trained model is neither overfitting nor underfitting. The model accuracy plots for both train and validation data are represented in Fig.12a. The model loss plots for both train and validation data are represented in Fig.12b. Training loss was only about 0.1 per cent, thus this model performs better and this implies that the trained model can perform accurate predictions based on images that will be fed to the system.

Lu et al. (2017) claimed that using their own convolutional neural network model, and they were able to achieve an accuracy of 95.48 per cent on a 500-leaf dataset of rice. Kulkarani (2018) reported that training data sets with 99.62 per cent accuracy of Mobile Net V2 model on the Plant Disease dataset, which is an open-source dataset that contains 54,306 photos of crop leaves classified into 38 different disease classifications. Ma et al. (2018) claimed that using their own convolutional neural network model, and they were able to achieve an accuracy of 93.4 per cent on 14,208 Cucumber leaf photos. Chen et al. (2019) claimed that using their own convolutional neural network model, and they were able to achieve an accuracy of 90.16 per cent.

### Table 3. Testing the image dataset

<table>
<thead>
<tr>
<th>Classification</th>
<th>Test dataset</th>
<th>True</th>
<th>False</th>
</tr>
</thead>
<tbody>
<tr>
<td>Healthy maize leaves</td>
<td>100</td>
<td>95</td>
<td>05</td>
</tr>
<tr>
<td>Pin hole caused by FAW</td>
<td>100</td>
<td>80</td>
<td>20</td>
</tr>
<tr>
<td>Circular hole caused by FAW</td>
<td>100</td>
<td>91</td>
<td>09</td>
</tr>
<tr>
<td>Ragged hole caused by FAW</td>
<td>100</td>
<td>98</td>
<td>02</td>
</tr>
<tr>
<td>Whorl damage caused by FAW</td>
<td>100</td>
<td>89</td>
<td>11</td>
</tr>
<tr>
<td>Nil damage to slight damage at tip of the cobs</td>
<td>100</td>
<td>91</td>
<td>09</td>
</tr>
<tr>
<td>&lt;25% of cob area showing FAW infestation</td>
<td>100</td>
<td>83</td>
<td>17</td>
</tr>
<tr>
<td>26-50% of cob area showing FAW infestation</td>
<td>100</td>
<td>87</td>
<td>13</td>
</tr>
<tr>
<td>51-75% of cob area showing FAW infestation</td>
<td>100</td>
<td>94</td>
<td>06</td>
</tr>
<tr>
<td>&lt;75% of cob area showing FAW infestation</td>
<td>100</td>
<td>96</td>
<td>04</td>
</tr>
<tr>
<td>Healthy maize tassel</td>
<td>100</td>
<td>98</td>
<td>02</td>
</tr>
<tr>
<td>Fall armyworm infested maize tassel</td>
<td>100</td>
<td>92</td>
<td>08</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>1200</strong></td>
<td><strong>1094</strong></td>
<td><strong>106</strong></td>
</tr>
</tbody>
</table>
cent on 3810 tea leaf photos. The present study employed its own convolutional neural network model, which achieved 100% training accuracy on a dataset of 11,251 images of fall armyworm infestation maize divided into 12 classes as the present deep learning model performed well.

**Learning rate analysis**

The learning rate is a hyper parameter. This reduces the model's inaccuracy by minimising the loss function. On a collection of 1088 ladies finger leaf image datasets, the exceptional accuracy of 94 percent was reached using CNN at a learning rate of 0.001 (Selvam and Kavitha, 2020). In this present study, at a learning rate of 0.00001, the outstanding accuracy of 100 percent was reached on the classification of maize fall armyworm infestation. It was found that as the learning rate was increased or decreased beyond the ideal rate of 0.00001, the accuracy decreased.

**Development of mobile application**

Hidayat et al. (2019) used CNN to construct an android app via android studio for maize corn diseases, using a total dataset of 3854 pictures of diseases in corn plants, including three different forms of corn diseases including Common Rust, Gray Leaf Spot, and Northern...

Leaf Blight. The accuracy in recognising disease in corn plants is 99 per cent.

In my research, the android application (AIPES) was built using the trained model with the assistance of Android Studio. It was appended to the system as a library. App interface of the developed android application is shown in Fig.13. The fall armyworm infested maize can be identified in two possibilities available in the application. One is to select the file, and the other is to use the camera. Images can either uploaded from the gallery where the test data was stored, or a camera can be used to shoot field photos of fall armyworm infested photos. A maize leaf, cob and tassel sample screen from the mobile application that uses the test dataset to classify the fall armyworm symptom as seen in Fig.14, Fig.15, Fig.16 respectively. The image was fed into the classifier, which was then matched against the maize fall armyworm infection database to get the

---

**Fig.12.** Accuracy and loss for train and validation dataset at 100 epochs. a) Model accuracy plot for train and validation dataset. b) Model loss plot for train and validation dataset.
Conclusion

The agricultural industry relies on early detection and recognition of these FAW infestations. In the present study, CNN was used to classify photos of maize affected by the autumn armyworm. Using images collected from maize research plots, the CNN architecture was trained. The present model had a training accuracy of 100% and a validation accuracy of 87% in terms of classification. This demonstrated that the CNN extracts essential information from photos with an unmanaged backdrop, which is necessary for classifying plant pest infestations. Experiments further revealed that the suggested CNN architecture could classify fall armyworm-infested maize into twelve categories. The number of datasets will be increased as part of this project’s extension as the study can boost the high accuracy in the validation.

Conflict of interest

The authors declare that they have no conflict of interest.

REFERENCES


