# Fall Armyworm (FAW) Infestation Project

Deep Learning Course 2019

Rafi Levy

ID: 057931354

levyrafi@mail.tau.ac.il

## I. Abstract

Automatic estimation of plant disease severity is essential for food security, yield-loss prediction and disease treatment recommendation. Thanks to deep learning methods, and in particular CNN (Convolutional Neural Networks), a huge progress has already been made in image processing. Since 2016 CNN models have been used in plant disease assessment for both binary (healthy/infested) and multiple (infestation severity) classifications. In this work we use such models to diagnose the damage caused by the Fall Armyworm (an insect) to maize crops. To our best knowledge, it is the first time such models are applied to maize images captured under uncontrolled conditions (pictures taken in a field). Using VGG16 (pre-trained on imagenet), we achieved 91.2% accuracy for binary classification, 68% for coarse severity classification, and 52.4% for fine-grained severity classification.

## II. Problem Description

Plant diseases are enormous problem for farmers. Fall Armyworm is a pest which has invaded to Africa a few years ago and severely attacks the maize crops, which are a major food source in that continent. As part of the efforts to track and battle the pest, a monitoring technique was proposed based on field scouting for estimating the level of infestation. This monitoring, being held in Kenya, is applied by volunteer farmers who provide photographs (taken by a smartphone) of most of the plants that have been examined. Specifically, for each field five samples of groups of plants (approximately 10 individual plants) were examined and photographed, and the number of infected plants was counted and reported. Due to differences in monitoring practices and lack of expertise, automatic tools for estimating and verifying the reliability of these reports are required. Such tools can be beneficial later for aerial photographing. Initially the aim of this research was to count the number of infested plants in a given image. However, that aim has been rephrased as estimating the infestation severity. By rephrasing the aim we assume a correlation between the level of infestation and the damage severity (manifested as tears or holes in the leaves). In short, we treat this problem as a classification task rather than object recognition.

## III. Dataset Description

The dataset consists of images captured under uncontrolled conditions (rather than in laboratory conditions). Some of the pictures are focused on a particular plant, but the majority capture a group of plants, namely reflect what the scout saw in the field (see Fig. 1). Unlike in DeChant et al. (2017), the lesions (the tears on the leaves) are not annotated. Obviously, the absence of these labels, the marking of damaged regions, challenges the model, which needs to fill in for this missing information.

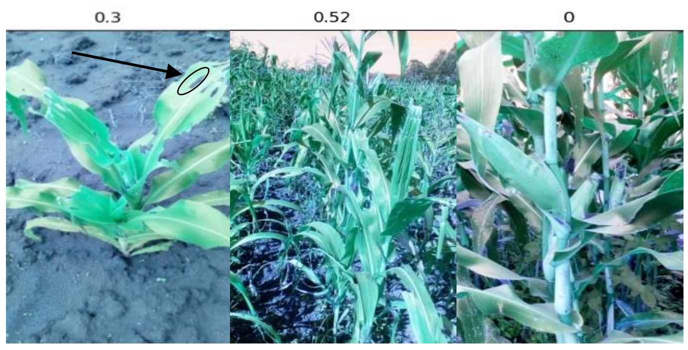

Fig. 1 – The left image is focused on a single plant image, while on the middle one we see a group of plants. On the right image we see a close-up of multiple plants. Numbers on top are the infestation levels. Tears on the leaves are visible in both left and middle images (the black arrow shows a particular tear on the left image), but the tears may appear more clearly in the individual plant image.

The dataset consists of pairs of a picture and its field ID and in addition a list of infestation levels (range between 0 and 1) attached to each field ID. Even though we were expecting to see five pictures or more from every field, in many cases the number was smaller than five. These cases were omitted from the dataset.

The number of pictures in the dataset is 13,231 most of which show healthy or lightly infected plants (Fig. 2).

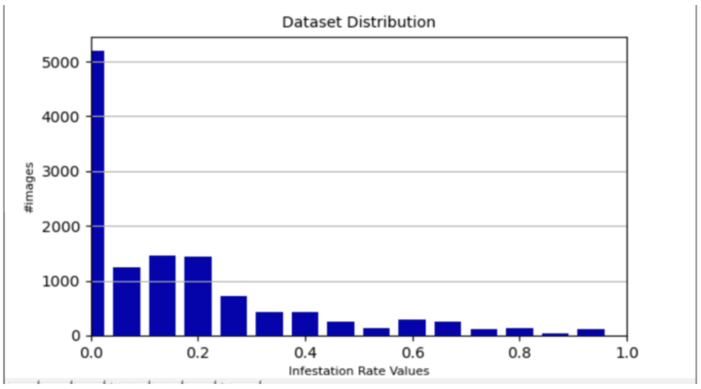

Fig. 2 - Distribution of pictures according to infestation level. Most of the pictures show healthy plants or plants with relatively low infestation level.

Most of the pictures are in vertical position, of size 1600x800, while the rest have the same dimensions but in horizontal position. During preprocessing images are resized to 512x512. As the data is new, baseline is not available. When dataset was partitioned to training, validation and test sets, we ensured images of the same field will not be shared by any two sets.

## IV. Architecture

We use a VGG16 network pre-trained on ImageNet dataset. On top of the last convolutional layer of VGG we add a 8x8 max pooling layer followed by two fully connected layers of 2048 and 1024 units with a ReLU activation. Each of the fully connected layers is followed by a dropout layer (with a dropout rate of 50%) and a batch normalization layer. The last fully connected layer has N outputs, corresponding to the number of classes, which feed into the softmax layer. Other architectures, such as ResNet50 and Inseption-V3, were explored. However, their performances were inferior to that of VGG16 (these findings are in line with Wang et al. 2017).

## V. Training Method

Initially we freeze the weights of the pre-trained model and we train the weights of the connected layers (up to 30 epochs). Next, we use fine-tuning – we unfreeze the weights of the last convolutional block of the VGG16 model and continue to train (up to 10 more epochs). We use Adam optimization with learning rate of 1e-3 for the first 10 epochs, 1e-4 for the next 10, and then 1e-5. Loss function is a categorical cross-entropy. Batch size is set to 32. In addition, we use early stopping and balanced class weights (to overcome the class imbalance problem). Weights which produce the maximum validation accuracy are saved on a disk and used for test prediction.

## VI. Experiments and Results

### A. *Binary classification*

We ran four experiments of binary classification. In the first one we labeled images with infestation level greater than 0 as Infested and images with 0 infestation level as Healthy. In the last three experiments we kept the Healthy images and divided the Infested ones into three classes, based on infestation severity: 0-0.3, 0.3-0.6, 0.6-1.0. We call these classes Level 1 Severity, Level 2 Severity and Level 3 Severity, respectively. We ran the model on Healthy vs. each of these severity classes. For the test data, we put aside 10% of the images to stay unused throughout the entire training procedure. The rest of the data is partitioned into training data (80%) and validation data (20%). Obviously we expected to see an increase in accuracy as the severity level becomes higher. Using the entire Infested set, accuracy is expected to be a number between the accuracy of Level 1 and the accuracy of Level 3 (see Fig. 3a, 3b, 3c and 3d). Accuracy results and ROC curves are depicted in Table 1 and in Fig. 4, respectively.

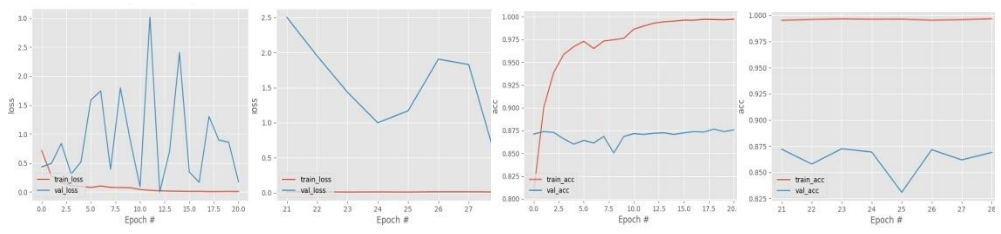

Fig. 3a - Training and validation loss/accuracy of Healthy vs. Infested. The left two show the loss during the pre-trained stage and the fine-tuning, respectively. The right two plots show the accuracy during these two stages.

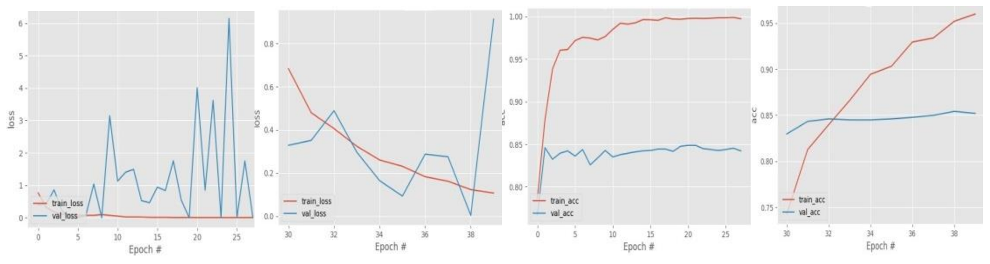

Fig. 3b - Training and validation loss/accuracy of Healthy vs. Level 1 Severity (0-0.3). The two left plots show the loss during the pre-trained stage and the fine-tuning, respectively. The right two plots show the accuracy during these two stages.

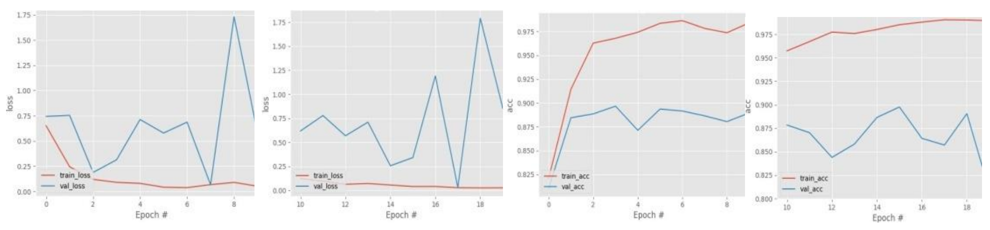

Fig. 3c - Training and validation loss/accuracy of Healthy vs. Level 2 Severity (0.3-0.6). The left two plots show the loss during the pre-trained stage and fine-tuning, respectively. The right two plots show the accuracy during these two stages.

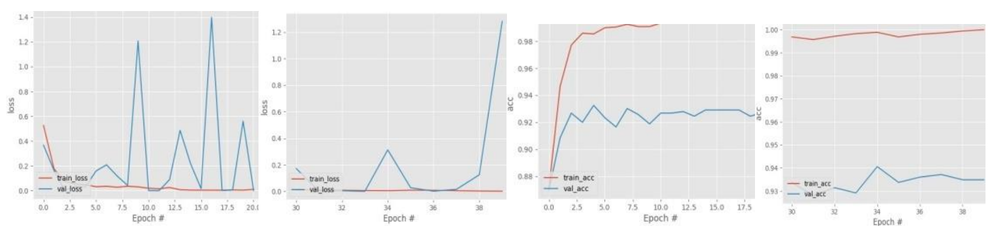

Fig. 3d - Training and validation loss/accuracy of Healthy vs. Level 3 Severity (0.6-1.0). The left two plots show the loss during the pre-trained stage and the fine-tuning, respectively. The right two plots show the accuracy during these two stages.

| Binary Experiment | Test Acc. |
|---|---|
| Healthy vs. Infested | 0.91 |
| Healthy vs. level 1 severity | 0.86 |
| Healthy vs. level 2 severity | 0.88 |
| Healthy vs. level 3 severity | 0.96 |

Table 1 – Test accuracy for each of the binary classification experiments. Test accuracy increases as the severity level gets higher.

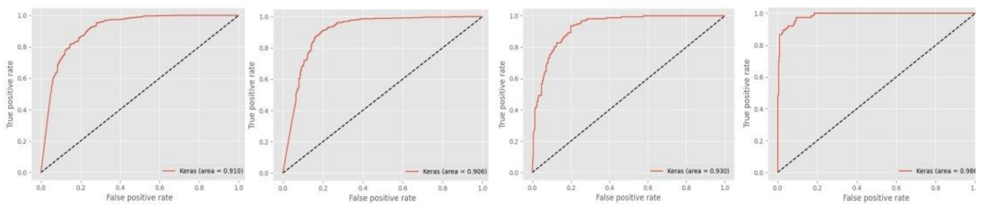

Fig. 4 – ROC curves of the binary classification experiments. From left to right: ROC of Healthy/Infested, ROC of Healthy/Level 1 Severity , ROC of Healthy/Level 2 Severity, the Healthy/Level 3 Severity. We can see the AUCs increasing as severity increases (as expected).

## B. *Multiple Classification*

We ran two experiments of multiple classes: coarse severity classification and fine-grained severity classification. In both we had a separate class for healthy plants. In the first experiment, we partitioned the diseased severity range into three intervals: 0-0.3, 0.3-0.6, 0.6-1.0. However, the boundaries of these intervals do not coincide. They are spaced on each side by a margin of 0.04, so samples which fall within these spaces are not used for training and validation. The severity intervals after applying the margin are therefore: 0.04-0.26, 0.34-0.56, 0.64-1.0. The second experiment is the same as the first one, but we partitioned the disease severity range into 5 intervals: 0-0.2, 0.2-0.4, 0.4-0.6, 0.6-0.8, 0.8-1.0. Margin of 0.04 was applied in the same manner. Partitioning the data into training, validation and testing sets was done in the same way as we did for the binary case. However, this time, for the testing data, we took an additional 10% from the data which fell in the spaces between the intervals. Obviously, this data is less distinguishable and can be mislabeled by the prediction. Indeed, we see a significant difference between test accuracy and validation accuracy, where the last is much higher. Fig. 5a and 5b show the results of the experiments. Accuracy results are depicted in Table 2.

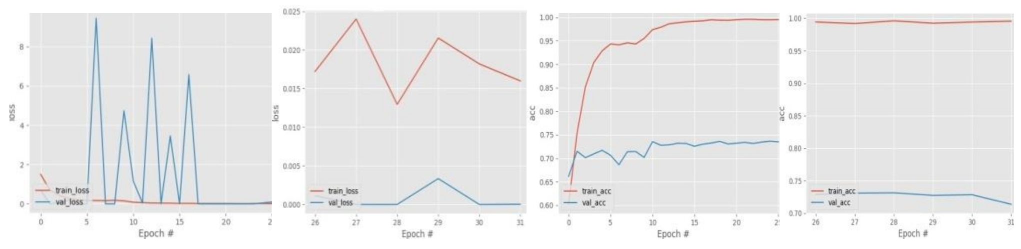

Fig. 5a - Training and validation loss/accuracy of coarse severity classification (4 classes). The left two plots show the loss during the pre-trained stage and the fine-tuning, respectively. The right two plots show the accuracy during these two stages. It is unclear how come validation loss is zero while accuracy is 74%.

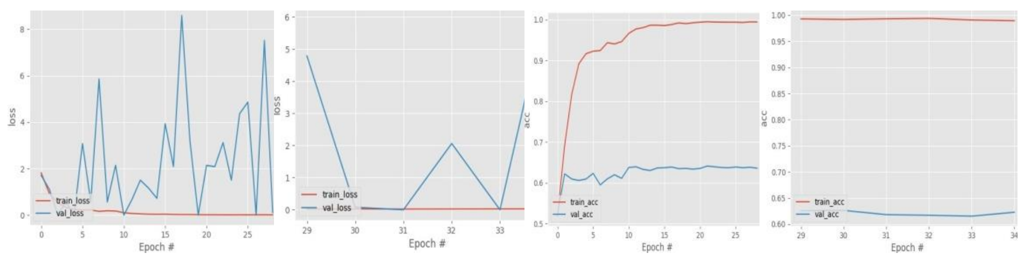

Fig. 5b - Training and validation loss/accuracy of fine-grained severity classification (6 classes). The left two plots show the loss during the pre-trained stage and the fine-tuning, respectively. The right two plots show the accuracy during these two stages.

| Multiple Classes Experiment | Test Acc. |
|---|---|
| Coarse Severity | 0.68 |
| Fine-grained Severity | 0.524 |

Table 2 - Test accuracy for each of the multiple classes experiments. Accuracy decreases as the number of classes increases.

| | 0.0 | 0-0.2 | 0.2-0.4 | 0.4-0.6 | 0.6-0.8 | 0.8-1.0 |
|---|---|---|---|---|---|---|
| 0.0 | 330 | 45 | 16 | 2 | 3 | 0 |
| 0-0.2 | 98 | 249 | 79 | 13 | 9 | 3 |
| 0.2-04 | 35 | 98 | 103 | 17 | 7 | 4 |
| 0.4-0.6 | 10 | 38 | 26 | 14 | 7 | 5 |
| 0.6-0.8 | 7 | 14 | 11 | 4 | 14 | 3 |
| 0.8-1.0 | 1 | 6 | 6 | 6 | 2 | 1 |

Table 3 – a confusion matrix of the 6 classes experiment. Prediction is good on the healthy, lightly and moderately infested plants (0-0.2 and 0.2-0.4), but degraded for the rest.

Comparing between test accuracy and validation accuracy (Fig. 5 and Table 2), we see that the gap between them increases as the number of classes increases. In order to better understand the accuracy results, we created a confusion matrix of the 6 classes experiment (see Table 3)

## C. *Visualization*

In order to see whether the model (of the binary classification Healthy/Diseased) is actually looking at the tears (holes) of the leaves, we generated Grad-CAM (Gradient Class Activation Mapping) heatmap of the final CONV layer. Fig. 6 shows four pairs of images, each pair consisting of the original image and the heatmap overlaid on top of that image. As we can see, it's not decisive to what extent the model is aware of the tears. In some cases though, where the camera was focusing on particular leaves, tears where colored by dark red (a warm color).

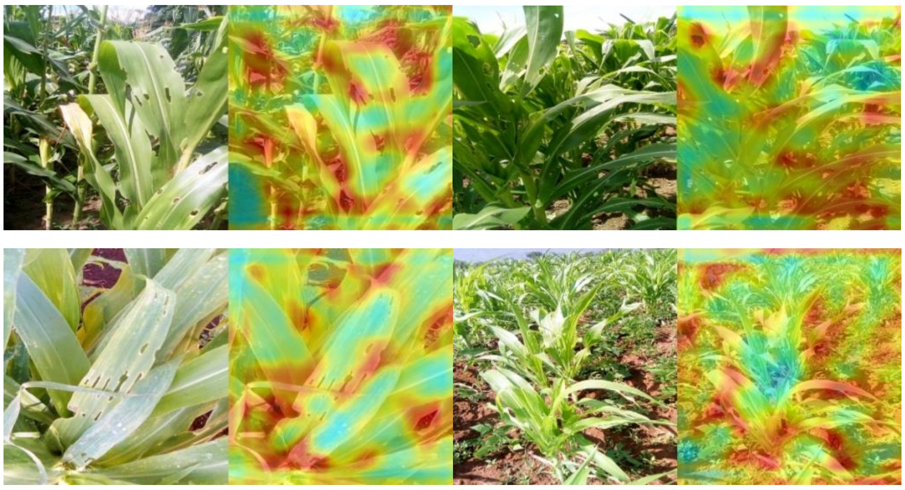
Fig. 6 – Four pairs of images, each pair consists of the original image and the Grad-CAM heatmap combined with it. We can see that the model is looking at the tears, at least in three of the images (the two upper pairs and the bottom left pair). In the bottom right pair, it's not really clear. The model seems to highlight regions where leaves and earth are near each other.

## VII. Code Explanation

Files are located in https://github.com/levyrafi12/DL_faw/tree/master/code.

The module faw.py contains the main and all the major functions: preprocessing, building the model, training and prediction. At the top of faw.py, there is a section of user defined parameters. These parameters control the batch size, the classification method (whether to run binary or multiple classifications), the enabling of training and which stage to train – freeze (pre-trained) or unfreeze (fine-tuning), the enabling of prediction-only mode (if training is disabled), the Grad-CAM visualization, etc. The module data_gen.py generates the next batch. The module grad_cam.py visualizes Grad-CAM heatmap of images from the training and the validation sets for binary classification model only. The module faw_utlis.py consists of utilization functions. The code uses the Keras library. Running command is python faw.py

## VIII. Ideas for Further Research

As the dataset is noisy, one possible direction would be to manipulate the dataset. We can partition the dataset to different groups and try to use different models for each group. We can extract the group of images of individual plants and train the model on that group only. We could do the same for images with lots of background (earth or sky), dried plants, plants which are not maize, etc. We could use the information about the stage of maize growth available in the report files and apply separate models for each growth stage.

Another direction is to treat this problem as MIL (multiple instance learning). We should remember that the infestation level is an attribute of the field and not of a single image. That means that each image can have a slightly different infestation level than the ground truth. However, the average level of the images of the same field should be close to that ground truth. I made a few steps in that direction. I treated the problem as a regression optimization by using the numerical infestation levels as labels, setting the metric to mean absolute error and changing the objective function (trying to bring not each predicted y, but rather the average of the predicted ys of the same field, closer to the ground truth. Unfortunately, initial results were not promising. Finally, we could perhaps initially train, from scratch, the PlantVillage dataset images (Hughes et al. 2015) taken under field conditions and then use that model as pre-trained for training our dataset (in short, replace imagenet by a subset of PlantVillage).

## IX. Summary

It is a difficult task to predict the infestation severity class of an image in this particular dataset. When looking at an image, the non-expert viewer would find it almost impossible to figure out which infestation severity class that image belongs to. It is not even easy to distinguish between healthy and infested plants (in low infestation level plants, the holes are sometimes hardly distinguishable). Since it is a new dataset, we didn’t have a baseline, so it's hard to tell whether results are disappointing or not. It's true that other works dealing with similar problems reached an accuracy of 95% and even more, but they were applied either to pictures taken in laboratory conditions or to annotated ones.

## X. Acknowledgement

I would like to thank Ofer Mendelson for offering this subject, being always available for questions, contributing valuable ideas and last but not least, helping in a pleasant way.

## XI. Bibliography

Boulent, J., Foucher, S., Theau, J., St-Charles, P. L., “Convolutional ´ neural networks for the automatic identification of plant diseases,” Frontiers in Plant Science, vol. 10, 2019.

Brahimi, M., Arsenovic, M., Laraba, S., Sladojevic, S., Boukhalfa, K., and Moussaoui, A. (2018). “Deep learning for plant diseases: detection and saliency map visualisation,” in Human and Machine Learning, eds J. Zhou and F. Chen (Cham: Springer International Publishing), 93–117.

DeChant, C., Wiesner-Hanks, T., Chen, S., Stewart, E. L., Yosinski, J., Gore, M. A., et al. (2017). Automated Identification of Northern leaf blight-infected maize plants from field imagery using deep learning. Phytopathology 107, 1426–1432.

Hughes, D. P., and Salathé, M. (2015). An open access repository of images on plant health to enable the development of mobile disease diagnostics.

Wang, G., Sun, Y., and Wang, J. (2017). Automatic image-based plant disease severity estimation using deep learning. Comput. Intell. Neurosci.

.