https://dx.doi.org/10.4314/dujopas.v11i3b.19
ISSN (Print): 2476-8316
ISSN (Online): 2635-3490
Dutse Journal of Pure and Applied Sciences (DUJOPAS), *Vol. 11 No. 3b September 2025*

Lightweight Squeeze U-Net for Pixel-Wise Classification of Cassava Anthracnose Disease on Edge Device

[1]Sodiq Kazeem , [2]Ajagbe Taofik , [3]Rufai Mohammed, [4]Saliu Lateef

[1,4]Department of Computer Engineering,
Yaba College of Technology,
Lagos,
Nigeria.

[2]Department of Computer Science,
Lagos State University Ojo,
Nigeria.

[3]Department of Computer Technology,
Yaba College of Technology,
Lagos,
Nigeria

Email: kazeem23@yahoo.com

Abstract

This paper introduces a lightweight deep learning model, a modified Squeeze U-Net for pixel-level segmentation of Cassava Anthracnose Disease on edge devices. Cassava, a critical crop for Nigeria, suffers significant yield losses due to anthracnose, which causes leaf distortions and spots. Traditional visual inspection methods are inefficient and error-prone, emphasizing the need for automated disease detection. Building on the architectures of U-Net and Squeeze U-Net, our approach incorporates innovative modifications that drastically reduce model complexity. The Modified Squeeze U-Net achieves a 48-fold reduction in size (7.11 MB compared to 386 MB for U-Net), and a significant decrease in parameter count while maintaining competitive accuracy (approximately 92–98%). Evaluations were performed using a dataset of 500 annotated cassava leaf images, with testing on a Raspberry Pi 3 demonstrating real-time, on-field detection without reliance on cloud infrastructure. Comparative analysis reveals that while U-Net provides the highest accuracy, its large size renders it impractical for resource-constrained environments, and Squeeze U-Net's efficiency comes at the cost of lower accuracy. In contrast, the Modified Squeeze U-Net strikes an optimal balance between performance and efficiency, making it particularly well-suited for precision agriculture applications. Future work will assess the model's effectiveness across different crops and its integration into portable devices such as smartphones.

Keywords: Deep Learning , Modified Squeeze U-Net, Pixel-wise Segmentation, Cassava Anthracnose , Edge Devices.

**Author for Correspondence*

Introduction

Cassava (*Manihot esculenta*) is a staple crop with significant economic potential, particularly in Nigeria, which produced nearly 59 million tons in 2017, making it one of the world's leading producers (FAO, 2022). Despite its importance, cassava production is threatened by various diseases (Priyadharshini, 2019; Toda and Okura,2019), including Cassava Anthracnose Disease (CAD), caused by *Colletotrichum* spp. The disease manifests as leaf distortion and irregular brown spots, negatively impacting yield and crop quality.

Traditionally, farmers rely on visual inspection for disease identification, a method that is time-consuming, labor-intensive, and prone to errors. In the era of digital transformation, deep learning offers an efficient alternative, enabling automated and precise identification of plant diseases. Sladojevic *et al*. (2016), Huang (2007), Price *et al*.(1993) and Ciresan et al. (2012) observed that the visual symptoms exhibited on cassava leaves affected by common diseases can be used to differentiate infections, whether assessed by human observation or deep learning algorithms. Advanced models can segment diseased regions, predict disease severity, and recommend appropriate treatments, ultimately improving agricultural productivity and reducing economic losses.

The advancement of deep learning, particularly Convolutional Neural Networks (CNNs), has revolutionized plant disease detection, offering more accurate and efficient diagnosis. Previous studies have demonstrated high classification performance using CNNs like AlexNet and GoogleNet (Mohanty et al., 2016; Ferentinos et al., 2018), while segmentation models like U-Net and DeepLabV3+ have enabled pixel-level disease localization (Gonçalves et al., 2021).

Efforts to improve deep learning efficiency for agricultural tasks have focused on optimizing models for deployment in resource-limited environments. For instance, Maryum et al. (2021) achieved improved classification using segmented cassava leaf images via U-Net and EfficientNet-B4. Beheshti and Johnson (2020) introduced the Squeeze U-Net, replacing traditional U-Net components with fire modules to reduce memory usage and computational cost, resulting in a 12-fold size reduction. Véstias et al. (2020), Wang et al. (2019) and Lobo Torres et al.(2020) emphasized the importance of compact models for edge computing, especially where internet connectivity is limited. However, while these models demonstrate effectiveness, many remain too large or computationally expensive for direct deployment on low-cost edge devices like the Raspberry Pi.

Despite advancements, a clear research gap persists in designing a lightweight, accurate, and real-time segmentation model specifically tailored for field deployment on edge devices in low-resource agricultural settings. This study addresses that gap by proposing a Modified Squeeze U-Net optimized for pixel-wise segmentation of Cassava Anthracnose Disease, ensuring minimal memory footprint and computational demand. Unlike previous works, this model is uniquely designed to operate independently of cloud infrastructure, allowing for on-site, real-time diagnosis using affordable devices like the Raspberry Pi 3. The novelty lies in the fusion of lightweight architecture with high segmentation accuracy, customized for cassava leaf disease detection .

METHODOLOGY

The system models employed comprises of U-Net, Squeeze U-Net and Modified Squeeze U-Net.

System Model

U-Net

The U-Net allows for quick and accurate picture segmentation.There are two components to the U-Net architecture as in figure 1 adjust the learning rates throughout the training process to enable quicker convergence and its improved management. Two consecutive 3x3 convolutions, a max-pooling layer, a ReLU activation unit, and so on comprise each block in the contracting route.

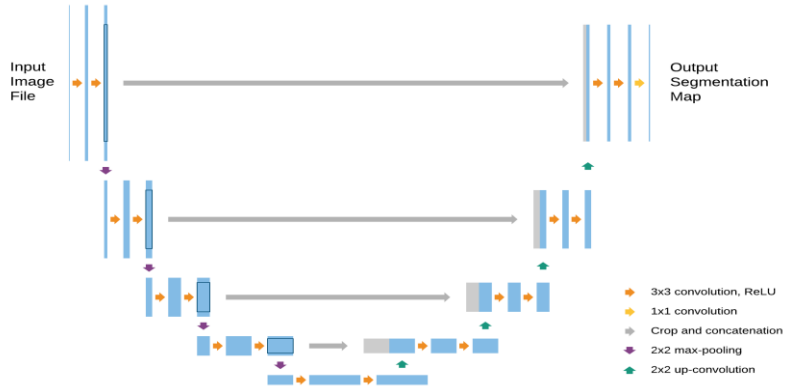

Figure 1: Basic U-Net Architecture (Agarwal *et al.*,2021)

Squeeze U-Net

The Squeeze U-Net design features downsampling units along the contracting path and upsampling units along the expansive path of the U-Net architecture as in figure 2.

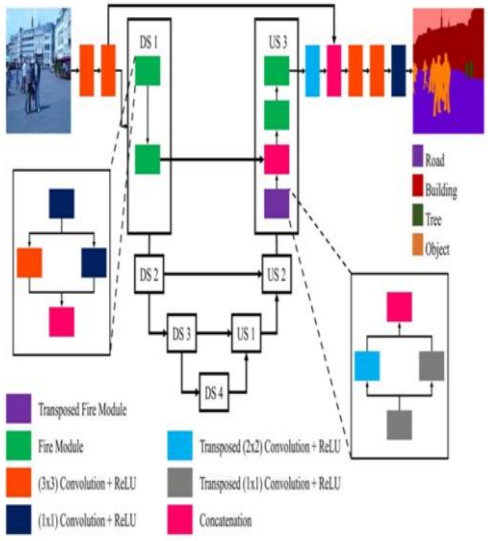

Figure 2: Squeeze U-Net

Each down-sampling (DS) unit comprises two fire modules that extract features. These features are then forwarded to both the subsequent DS unit and the corresponding up-

sampling (US) unit. In each up-sampling unit, a transposed fire module, a concatenation unit, and two additional fire modules work together to up-sample the input, extract features, and merge them to form the output.

Drawing inspiration from SqueezeNet, fire modules were incorporated into the contracting path of the Squeeze U-Net to facilitate downsampling. Each fire module in this path comprises a 1×1 convolution layer that outputs C'0 channels (where C'0 is less than C'i), followed by an inception-style layer that applies two parallel convolutions, one with a 3×3 kernel and the other with a 1×1 kernel, each generating C'i/2 channels. The outputs from these two convolutions are concatenated to form the final output of the fire module. This output is then forwarded to the next layer in the contracting path and linked to its corresponding layer in the expansive path through long skip connections. To improve the model's representational capacity, stride-2 convolutions are used for downsampling in place of traditional max or average pooling. Furthermore, employing fire modules in the expansive path contributes to a reduction in overall parameter count. The size of the squeeze U-Net is relatively small i.e 32MB compared to the size of the original U-Net model i.e 386MB .The squeeze U-Net was selected not just because of the model size, but with critical consideration of the parameters and kernel size.

Modified Squeeze U-Net

The modified U-Net was employed to reduce the model size and increasing the computational speed. In the original Squeeze U-Net model, there are upsampling units and downsampling units each containing fire modules; containing a rectangular arrangement of two (1x1)Conv2d layers, a (3x3)Conv2d layer and a Concatenation layer as illustrated in figure 3. For the modification, the fire modules were kept for both up sampling and down sampling units but rather than the recurrent layers of fire modules in the up sampling and down sampling units, there will only be a single layer of fire module.

The modified Squeeze U-Net (Figure 3) architecture includes both downsampling and upsampling units, similar to the structure of the original Squeeze U-Net.

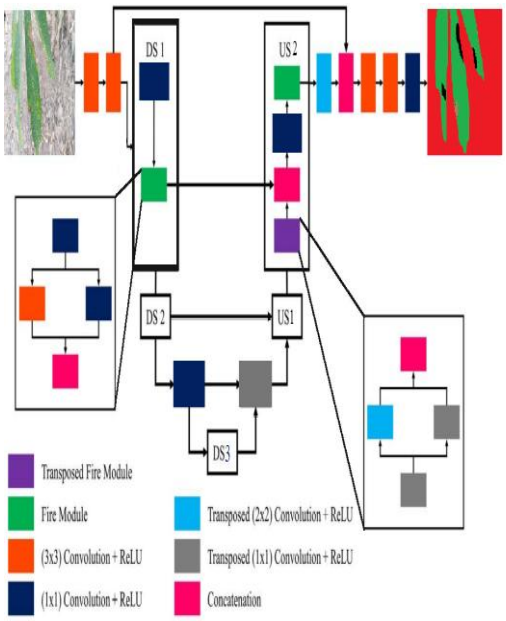

Figure 3: Modified Squeeze U-Net

Each downsampling (DS) unit is composed of a single fire module responsible for feature extraction. These extracted features are then forwarded to both the subsequent downsampling

unit and the matching upsampling (US) unit. In the upsampling process, each unit includes a transposed fire module, a concatenation block, and another fire module. Together, these components perform upsampling, extract relevant features, and merge feature maps to generate the final output.

To reduce the models size at a negligible cost of accuracy, the number of down sampling and up sampling units were reduced by replacing bounded up sampling and down sampling unit with a (1x1) convolution layer and a transpose (1x1) convolution layer.

Experimental Setup

Experiments were conducted using Hp Laptop with Intel core i5 CPU 2.5GHz, 8GB RAM on 64-bit Windows pro. Python programming language was used for implementation. Deep learning tasks were implemented using TensorFlow and Keras.

The disease spots of Cassava Anthracnose leaf diseases was identified using three (3)segmentation-based models.The dataset for the research was collected at Pava farm, Alawe farm, kanranjangan farm, bode osi farm, sekan farms in Osun state, Nigeria. A total of 500 leaf images of Cassava Anthracnose was collected for training, testing and validation. An infinix hot 8 mobile device camera was used to manually collect the data..Each image was taken in the *JPEG format and with an average image size of 2.8mb which resulted in an original 3 dimensional image array of the size (replace with image size).The dataset was also preprocessed. APEER web-based software was used to annotate the dataset. The Dataset was splitted with into a ratio of 50% for the training, 30% for the Test set and 20% for validation.

The figure 4 and 5 shows the images of leaves on annotation software before and while the leaf is being annotated.

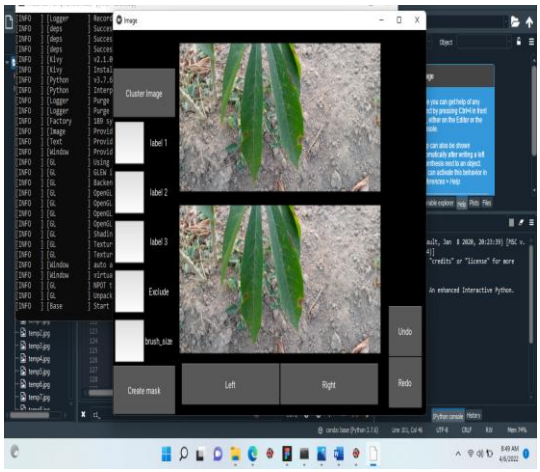

Figure 4: Original Cassava Leaf Image

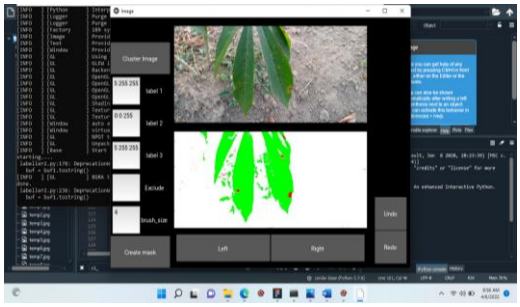

Figure 5: Original Cassava Leaf Image Being Annotated

Augmentation techniques such as Horizontal flip, vertical flip etc were applied to the training set. These techniques increase dataset for training by generating modified images artificially (Ghosal *et al.*, 2018). Thus, they make prevention of overfitting and enhancement of model performance possible (Pawara *et al.*, 2017; Barbedo, 2018) .

Implementation

Hardware Implementation:
As outlined by Trickiknow (2018), the Raspberry Pi's GPIO pins are integrated with a TFT touchscreen to enable display functionality. A 32GB SD card is inserted into the Raspberry Pi to serve as storage, while a micro USB cable is used to supply power.
The Raspberry Pi features a single HDMI port for display output. However, for older monitors that only support VGA, a VGA-to-HDMI adapter can be used to establish a connection via the VGA port.

Software Implementation:
The Raspbian OS was installed on the Raspberry Pi following these steps:

1. Download Raspbian OS from the official Raspberry Pi website: https://www.raspberrypi.org/downloads/raspbian/ (Ensure to download the full desktop version).
2. Extract the OS image file using 7-Zip, which can be downloaded from: http://www.7-zip.org/.
3. Insert a Micro SD card into a PC or laptop using a card reader.
4. Download and install an image flashing tool, such as Etcher, from: https://etcher.io/.
5. Open Etcher, select the extracted Raspbian OS image file, choose the Micro SD card as the target storage, and click Flash (Figure 6) to complete the installation.

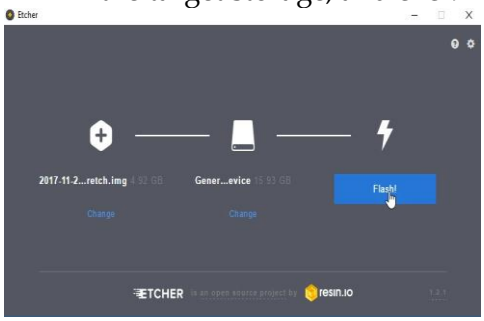

Figure 6: Etching

Step 6: Once the flashing process is complete, remove the memory card from the PC and insert it into the SD card slot on the Raspberry Pi(Figure 7).


Figure 7: Inserting Raspberry Pi SD Card

Step 7: The Raspbian OS was successfully installed on the Raspberry Pi. Next, a keyboard and mouse were connected to the USB ports of the Raspberry Pi, while a PC monitor was linked to the device using an HDMI cable.
Installation of LCD Display on Raspberry Pi and software : This requires connecting the LCD to the GPIO pins on Raspberry Pi (figure 8).

Figure 8: Installing LCD Display on Raspberry Pi

The software was installed and the screen displayed (figure 9).


Figure 9: Installed Raspberry Pi LCD

RESULTS AND DISCUSSION

The dataset was trained as shown in Figure 10, using the developed Modified Squeeze U-Net, Squeeze U-Net, and U-Net architectures. Hyperparameter tuning was performed using a random search strategy, which, compared to grid search, converges on optimal parameters with fewer iterations. The input image size used was 1872 × 4160 pixels. Squeeze U-Net achieved a 2× reduction in training time compared to U-Net, albeit with a drop in accuracy. To enhance training performance, a learning rate (LR) scheduler was employed to reduce the learning rate when no improvement in loss was observed after five epochs.

Figures 10, 11, and 12 respectively present the training-validation screenshot, training and validation loss, and training and validation accuracy curves, providing a visual understanding of the models' learning behavior across epochs.

```
Epoch 1/30
100/100 [==============================] - 203s 2s/step - loss: 0.5811 - accuracy: 0.7626 - val_loss: 46.1589 - val_accuracy: 0.4155
Epoch 2/30
100/100 [==============================] - 166s 2s/step - loss: 0.3364 - accuracy: 0.8742 - val_loss: 5.7541 - val_accuracy: 0.4279
Epoch 3/30
100/100 [==============================] - 175s 2s/step - loss: 0.2794 - accuracy: 0.8977 - val_loss: 2.4677 - val_accuracy: 0.5244
Epoch 4/30
100/100 [==============================] - 166s 2s/step - loss: 0.2530 - accuracy: 0.9081 - val_loss: 2.8215 - val_accuracy: 0.4759
Epoch 5/30
100/100 [==============================] - 166s 2s/step - loss: 0.2381 - accuracy: 0.9134 - val_loss: 1.8062 - val_accuracy: 0.5991
Epoch 6/30
100/100 [==============================] - 175s 2s/step - loss: 0.2261 - accuracy: 0.9179 - val_loss: 0.6891 - val_accuracy: 0.7995
Epoch 7/30
100/100 [==============================] - 175s 2s/step - loss: 0.2101 - accuracy: 0.9243 - val_loss: 0.2602 - val_accuracy: 0.9086
Epoch 8/30
100/100 [==============================] - 167s 2s/step - loss: 0.2023 - accuracy: 0.9265 - val_loss: 0.2164 - val_accuracy: 0.9235
Epoch 9/30
100/100 [==============================] - 167s 2s/step - loss: 0.1943 - accuracy: 0.9297 - val_loss: 0.1878 - val_accuracy: 0.9345
Epoch 10/30
100/100 [==============================] - 175s 2s/step - loss: 0.1793 - accuracy: 0.9354 - val_loss: 0.1969 - val_accuracy: 0.9308
Epoch 11/30
100/100 [==============================] - 167s 2s/step - loss: 0.1770 - accuracy: 0.9363 - val_loss: 0.2048 - val_accuracy: 0.9245
```

Figure 10: A Screenshot of Training - validation accuracy during a couple of epochs

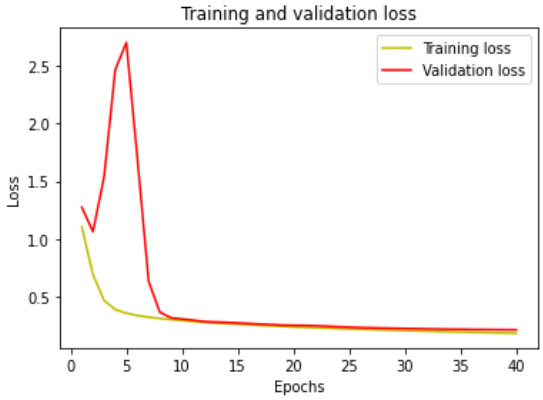

Figure 11: Training and Validation Loss

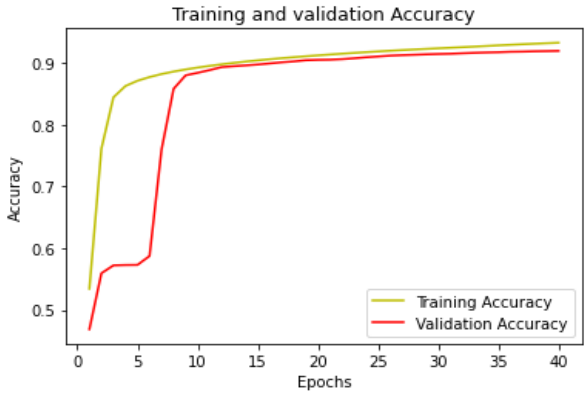

Figure 12 : Training and Validation Accuracy

The models were evaluated on cassava and yam leaf disease datasets using key performance metrics, including parameter count, model size (in MB), kernel types, and accuracy (%). These metrics are summarized in Scientific Table 1.

Table 1: Scientific Table

Model Architecture	#Params	Mode Size (MB)	Kernel	Accuracy (%)
Modified Squeeze U-Net	1.51M	7.11	3 × 3 2 × 2 1 × 1	92.15
Squeeze U-Net	2.59M	32	3 × 3 2 × 2 1 × 1	85.91
U - Net	30M	386	3 × 3 2 × 2 1 × 1	97.81

The table compares the three segmentation models i.e Modified Squeeze U-Net, Squeeze U-Net, and U-Net—based on performance and efficiency. U-Net, with 30 million parameters and a 386 MB size, achieved the highest accuracy (97.81%), consistent with findings by Gonçalves et al. (2021) and Maryum et al. (2021), who noted U-Net's robustness in high-fidelity semantic

segmentation tasks. However, the trade-off lies in its computational cost, making it unsuitable for real-time, edge-level deployment.

Squeeze U-Net, introduced by Beheshti and Johnson (2020), shows a marked improvement in resource efficiency, reducing model size to 32 MB with a moderate accuracy of 85.91%. This efficiency gain is attributed to the use of fire modules and point-wise convolutions, as also reported in their work.

The Modified Squeeze U-Net, developed in this study, further optimizes the architecture by reducing the parameter count to 1.51 million and size to 7.11 MB, while maintaining a strong accuracy of 92.15%. This performance suggests a significant recovery in accuracy compared to standard Squeeze U-Net, while sustaining minimal computational overhead. This finding aligns with trends highlighted in Véstias et al. (2020), who emphasized the need for edge-efficient architectures for agricultural applications.

These results highlight a clear trade-off between model size and predictive performance. While U-Net provides the best accuracy, its size limits its usability in constrained environments. The Squeeze U-Net lowers complexity but sacrifices performance. The Modified Squeeze U-Net achieves an effective middle ground, suitable for deployment on resource-limited platforms like the Raspberry Pi, which is increasingly relevant for rural and field-based agricultural monitoring.

The architectural consistency in kernel sizes across the models (3×3, 2×2, 1×1) confirms that improvements in efficiency and accuracy stem from structural innovations, not from altered convolutional operations. The novelty of this research lies in its tailored architectural refinement of the Squeeze U-Net to meet the dual demands of accuracy and lightweight design for plant disease detection on edge devices.

CONCLUSION

This research focuses on the design and implementation of a modified Squeeze U-Net for segmenting cassava anthracnose leaf disease on an edge device. The optimized Squeeze U-Net model is highly lightweight, with a size of approximately 7.11 MB, achieving a 48-fold reduction compared to the original U-Net. The model was tested on a Raspberry Pi 3 to identify diseased regions in cassava leaves, yielding a high prediction accuracy of around 98%. It is suggested that future research explore the model's effectiveness in detecting diseased spots in various crops and assess its deployment on edge devices, such as smartphones, to further validate its performance.

REFERENCES

Agarwal, M., Gupta,S., Biswas, K. Plant Leaf Disease Segmentation Using Compressed U-Net Architecture.PAKDD Workshops, LNAI 12705, pp. 9–14, 2021. https://doi.org/10.1007/978-3-030-75015-2_2.

Beheshti, N. and Johnson,L. Squeeze U-Net : A memory and energy efficient image segmentation network .Paper Presented at CVPR 2020 workshop, 2020.

FAO 2014, Nigeria Agriculture at a Glance, Nigeria, accessed 20 February 2016, https://www.fao.org/nigeria/fao-in-nigeria/nigeria-at-a-glance/fr/

Ferentinos, K.P. Deep learning models for plant disease detection and diagnosis. Comput. Electron. Agric. 2018, 145, 311–318.

Juliano P. Gonçalves, Francisco A.C. Pinto, Daniel M. Queiroz, Flora M.M. Villar, Jayme G.A. Barbedo, Emerson M. Del Ponte, Deep learning architectures for semantic

segmentation and automatic estimation of severity of foliar symptoms caused by diseases or pests,Biosystems Engineering,Volume 210, 2021,Pages 129-142, ISSN 1537-5110,https://doi.org/10.1016/j.biosystemseng.2021.08.011.

Lobo Torres, D., Queiroz Feitosa, R., Nigri Happ, P., Elena Cué La Rosa, L., Marcato Junior, J., Martins, J., Olã Bressan, P., Gonçalves, W.N., Liesenberg, V. Applying fully convolutional architectures for semantic segmentation of a single tree species in urban environment on high resolution UAV optical imagery. Sensors 20:563, 2020.

Maryum,A.,Akram, M.U. and Abdul Salam, A. Cassava Leaf Disease Classification using Deep Neural Networks.Proceeding of IEEE 18th International Conference on Smart Communities: Improving Quality of Life using ICT, IoT & AI (HONET 2021), 2021.

Mohanty, S.P.; Hughes, D.P.; Salath, M. Using deep learning for image-based plant disease detection. Front. Plant Sci. 2016, 7, 1419.

Véstias, M.P., Duarte,R.P., Sousa, J.T. and Neto, H.C. Moving Deep Learning to the Edge. Algorithms 2020, 13, 125; doi:10.3390/a13050125.

Wang, T., Rostamza, M., Song, Z., Wang, L., McNickle, G., Iyer-Pascuzzi, A.S., Qiu, Z., Jin, J. SegRoot: A high throughput segmentation method for root image analysis. *Comput. Electron. Agric.* 162:845–854, 2019.

Zakir H. Raspberry Pi 3 Complete Tutorial, India Pundag, accessed 20 January 2018 from https://trickiknow.com/terms-and-conditions/>\