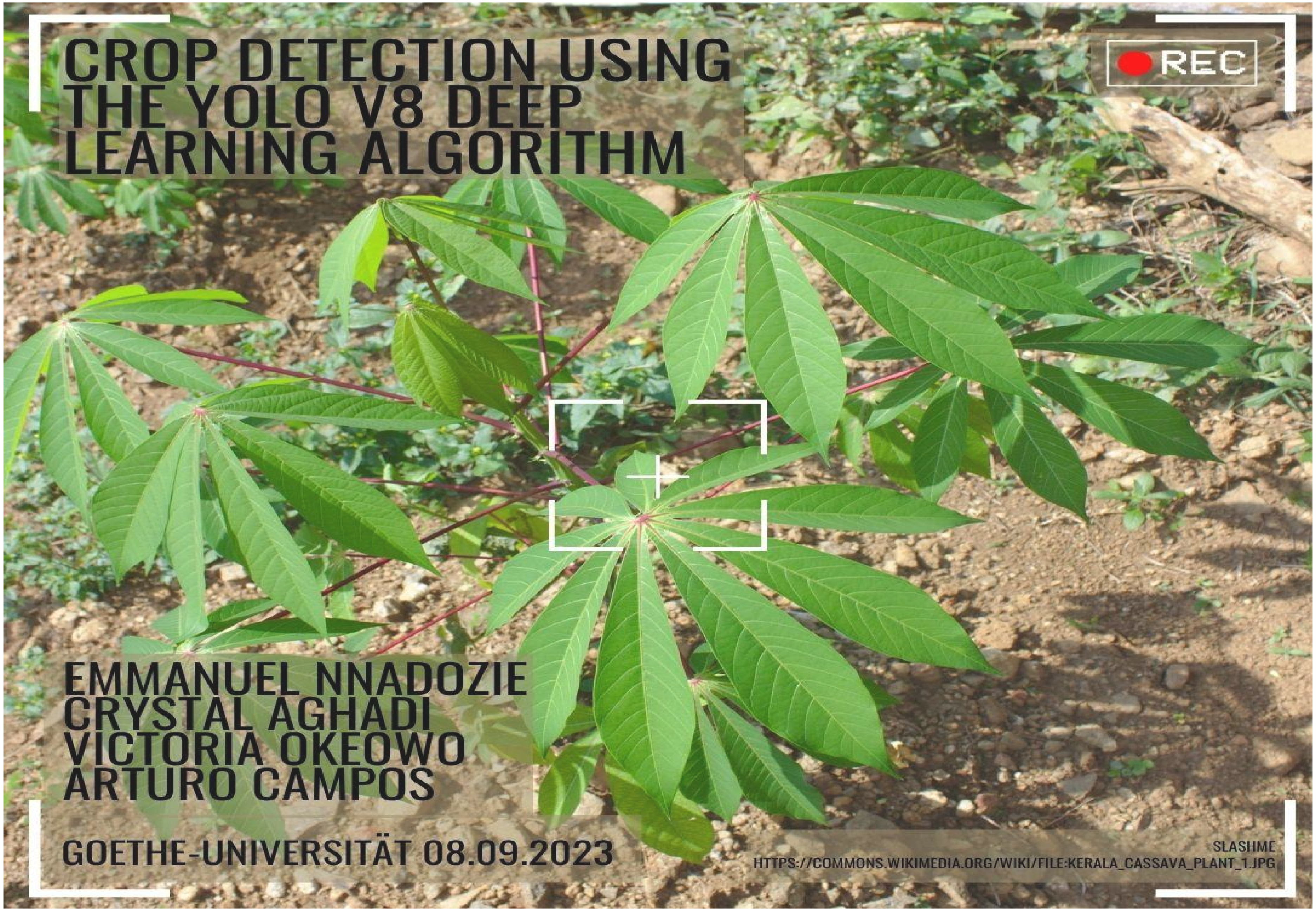

CROP DETECTION USING THE YOLO V8 DEEP LEARNING ALGORITHM

EMMANUEL NNADOZIE
CRYSTAL AGHADI
VICTORIA OKEOWO
ARTURO CAMPOS

GOETHE-UNIVERSITÄT 08.09.2023

SLASHME
HTTPS://COMMONS.WIKIMEDIA.ORG/WIKI/FILE:KERALA_CASSAVA_PLANT_1.JPG

Crop Detection using the YOLOv8 Deep Learning Model

Group members: *Emmanuel Nnadozie, Crystal Aghadi, Victoria Okeowo, and Arturo Campos*

Section 1: Introduction to the learning unit

Title	Crop Detection using the YOLOv8 Deep Learning Model
Duration	2 days (5 hours per day)
Introduction	Welcome to a tutorial guide on crop detection using the YOLOv8 deep learning model. Crop detection has numerous applications in precision agriculture, such as crop counting, weed detection, disease detection, disease control, and many more. Crop count, for example, can help determine germination rate, which can provide better information on plant vigor and serve as a basis for better yield estimation. However, it could be quite hectic to manually perform this process. The good news is that the process of crop detection can be automated using computer vision techniques like YOLOv8, which are both accurate and time-saving. Here, we will teach you how to use the YOLOv8 Deep Learning Model to detect crops in the field from a dataset of images captured by an Unmanned Aerial Vehicle (UAV). Do not worry if you have little or no prior experience with programming. This tutorial will guide you through a step-by-step process on how you can use the YOLOv8 deep learning model to detect crops

	using the lines of code provided in the tutorial. Our crop of interest in this tutorial is Cassava (*Manihot esculenta*). Following this tutorial, you should be able to successfully train the YOLOv8 model to accurately predict any plant (in this case, cassava) using image data.
Learning outcomes	*After completing the learning unit, you will be able to:* ● *Explain the concept of crop detection with deep learning* ● *Create annotated images and prepare data for training the YOLOv8 deep learning models* ● *Use Google Colab environment for YOLOv8 model training* ● *Train and validate YOLOv8 detection model for crop detection* ● *Predict cassava plants in new images using the YOLOv8 trained model*
Material	Please have the following items on hand for this tutorial: ● A computer and good Internet connection: You will need to install these software and packages on your PC if you have not already (more details on how to do this will be shown in this tutorial): ○ *If you use Windows: Python3 + Qt5 + install lxml or Anaconda* ○ *If you use Linux/UNIX: Python3 + Qt5* ○ *If you use macOS:* Python3 + *Qt5*

	● RGB images of a crop farm taken with UAV ○ We have provided RGB images of a cassava farm that you can use as a sample: click here. ("~\Group 6 content development\Annotation_data")
Literature	● *Nnadozie, E. C., Iloanusi, O. N., Ani, O. A., & Yu, K. (2023). Detecting Cassava Plants under Different Field Conditions Using UAV-Based RGB Images and Deep Learning Models. Remote Sensing, 15(9), 2322. https://doi.org/10.3390/rs15092322* ● *Rashid, J., Khan, I., Ali, G., Almotiri, S. H., AlGhamdi, M. A., & Masood, K. (2021). Multi-Level Deep Learning Model for Potato Leaf Disease Recognition. Electronics, 10(17), 2064. https://doi.org/10.3390/electronics10172064* ● *Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016 (pp. 779–788).*
Metadata	*Name: Crop Detection using the YOLOv8 Deep Learning Model* *Learning Resource Type: Tutorial* *Description: Crop Detection using the YOLOv8 Deep Learning Model* *Keyword(s): YOLOv8, Crop detection, Python* *Author(s): Emmanuel Nnadozie, Crystal Aghadi, Victoria Okeowo, and Arturo Campos* *Language: English* *License: CC Attribution-Noncommercial-Share Alike 4.0 International* *Creation Date: September 8, 2023*

Section 2: Input

Plant detection is an important step in farm management tasks like crop counting, yield estimation, and germination determination. Early crop detection methods used traditional image processing techniques that were not very accurate. Machine learning techniques were then used for crop detection, but they required time-consuming feature engineering techniques. Deep learning-based detection techniques that are more efficient have been developed.

The YOLOv8 object detection model is a free and open source deep learning model that employs a single-stage detection approach. YOLOv8 has been used for crop detection and has demonstrated good accuracy and speed. The application of YOLOv8 for crop detection consists of two stages: data collection and preparation, and model training, validation, and prediction rates.

Section 3: Application

Here, we will walk you through the steps of data preparation, model training, validation, and prediction.

1. **Data preparation:** To prepare your data, follow the steps below. You can follow along with the screencast guide here: ("~\Group 6 content development\01_Image annotation tutorial-08.09.2023_10.57.45_REC.mp4")

➢ Data download: You can download our data here, or you can collect your own data, which should include images of one or more types of crops taken with a UAV drone. Please also ensure that the data is in the YOLO format by specifying the training and validation data. You can explore our data to see the format and folders needed to arrange your data
If you are using our data, please prepare it as described below.
   - Download the zip folder here
   - Unzip the folder

- ➢ Installations: install python3, pyQt5, lxml, and labelimg
  - Install Python 3 using the links provided above (make sure to select pip installation when installing Python 3; you can install pip differently using the links provided in the appendix).
  - Navigate to the labelimg.tzutalin Github page.
  - Install the pyQt5, and lxml packages by following the instructions for your computer type.
  - Download the entire code for labelimg.tzutalin by clicking on the arrow next to the code tab in the top right corner of the screen.

- ➢ Change the class
  - Open the zip folder you downloaded from Github.
  - Navigate to the ‘data’ folder, where you will see a text file called “predefined class”.
  - Edit the classes there to the classes that you intend to define
  - In our case, we will delete the text there and type-in “cassava” as the only class because cassava is our crop of interest for the purpose of this tutorial.

- ➢ Run the labelimg.tzutalin package
  - Unzip the file you downloaded and copy the path
  - Open the command prompt on your PCs and set the working directory to the path, which you have copied: you can set the directory simply by typing cd, followed by the path you copied and click on ‘enter’
  - Run the next line of command in Github ‘`pyrcc4 -o libs/resources.py resources.qrc`’
  - Run the second line of command ‘For pyqt5, pyrcc5 -o libs/resources.py resources.qrc’
  - On the next line type ‘python labelimg.py’ and enter or run it
  - A window pops-up after running this command

- ➢ Browsing the labelImg Graphics User Interface
  - Open the directory to set your path to the training images ("~/Annotation_data/data/train/images").
    NOTE: Once you do this, the images will be automatically imported into the platform. The bottom right panel shows you the list of all the images that you have successfully uploaded)
  - Set the directory for saving your annotated images by clicking on 'Change Save Dir' and selecting the 'labels' folder under the train data ("~/Annotation_data/data/train/labels")
  - Make sure the image format is set to 'YOLO', you will see a tab written as 'YOLO' on the left side panel. This is the eight tab on the left panel

- ➢ Annotating the images
  - Click on 'Create Rectbox'
  - Then click and drag to draw bounding boxes around the cassava plants
  - Save it to the cassava class
  - Do this for all the images

2. Model Training, Validation and Prediction Rates

The second tutorial screencast will take you through the steps for model training, validation, and prediction
A completely annotated dataset has been provided for you.

Section 4: Assessment and Wrap up

Now that you have gone through this crop detection course, you can now try your hands on these assessments

- Could you explain why deep learning outperforms previous crop detection methods?
- Can you say what would happen if your annotation bounding boxes were not drawn tightly around the objects of interest?
- Explain briefly how using GPUs on Google Colab improves the model training process.
- How would increasing the confidence threshold affect cassava prediction on new images?
- Let us take a look at your model's performance. What is the mean average precision (mAP) of your trained YOLOv8 model?

  Your mAP should not be less than 0.85.
- How can you improve performance if your mAP is less than 0.85?
- Reflect on the quality and simplicity of the YOLOv8 model for crop detection

***Note** that you can apply the steps in this course to other plant images.*

Section 5: Appendix

Materials:

- *Tutorial Screencast 1: Dataset preparation and Image Annotation:* ("~\Group 6 content development\01_Image annotation tutorial-08.09.2023_10.57.45_REC.mp4")
- *Tutorial Screencast 2: YOLOv8 model training, validation, and inference:* ("~\Group 6 content development\02_YOLOv8 training tutorial_model training.mp4")
- *Dataset for annotation*
- *Annotated data for model training, validation, and inference*
- *YOLOv8 model training notebook*

Installations

- ***If you use Windows***

  Python3 + Qt5 + install lxml: *Open cmd and go to the labelImg directory*

  pyrcc4 -o libs/resources.py resources.qrc

  For pyqt5, pyrcc5 -o libs/resources.py resources.qrc

  python labelImg.py

  python labelImg.py [IMAGE_PATH] [PRE-DEFINED CLASS FILE]

  or

Anaconda: *Open the Anaconda Prompt and go to the labelImg directory*

conda install pyqt=5

conda install -c anaconda lxml

pyrcc5 -o libs/resources.py resources.qrc

python labelImg.py

python labelImg.py [IMAGE_PATH] [PRE-DEFINED CLASS FILE]

- ***If you use Linux/UNIX***

  Python3 + Qt5

  sudo apt-get install pyqt5-dev-tools

  sudo pip3 install -r requirements/requirements-linux-python3.txt

  make qt5py3

  python3 labelImg.py

  python3 labelImg.py [IMAGE_PATH] [PRE-DEFINED CLASS FILE]

- ***If you use macOS***

  Python3 + Qt5

  brew install qt # Install qt-5.x.x by Homebrew

  brew install libxml2

  or using pip

  pip3 install pyqt5 lxml # Install qt and lxml by pip

```
make qt5py3
python3 labelImg.py
python3 labelImg.py [IMAGE_PATH] [PRE-DEFINED CLASS FILE]
```

- ***Pip (Package installer for Python)***

You can use pip to install packages from the Python Package Index and other indexes.

Usually, pip is automatically installed if you are:

1. *Working in a virtual environment.*
2. *Using Python downloaded from python.org.*
3. *Using Python that has not been modified by a redistributor to remove ensurepip.*

If your Python environment does not have pip installed, there are 2 mechanisms to install pip supported directly by pip's maintainers:

a. ***ensurepip***

Python comes with an ensurepip module, which can install pip in a Python environment.

If you use Windows

```
C:> py -m ensurepip --upgrade
```

If you use Linux/UNIX

```
python -m ensurepip --upgrade
```

If you use macOS

```
python -m ensurepip --upgrade
```

b. get-pip.py

This is a Python script that uses some bootstrapping logic to install pip. Then, open a terminal/command prompt, cd to the folder containing the get-pip.py file and run.

If you use Windows

```
C:> py get-pip.py
```

If you use Linux/UNIX

```
python get-pip.py
```

If you use macOS

```
python get-pip.py
```