University of East London

Ain Shams University
Faculty of Engineering

Date: 22 / 5 / 2024

CSE485: Deep Learning 🤖

Egyptian Currency Real-Time Detection

By:

Omar Mosaad Atalla Abdellatif Diab

20P3176

Submitted to:

Dr. Alaa Hamdy

Contents

1) Introduction

I made an application that uses **YoloV8 model** that I trained on Egyptian Currencies detect Egyptian currencies in **real-time** using a camera feed. (open-cv)

GitHub Repository: -
https://github.com/OmarMDiab/Egyptian_Currency_Detection

2) Features

1) **Real-Time Detection:** The App opens the camera feed and continuously detects Egyptian currency notes within the frame.

See this Gif ➔ Accurate Bounding Boxes and Detects both Sides!

2) **Total Money Display:** It displays the **total sum of money** detected in a frame.

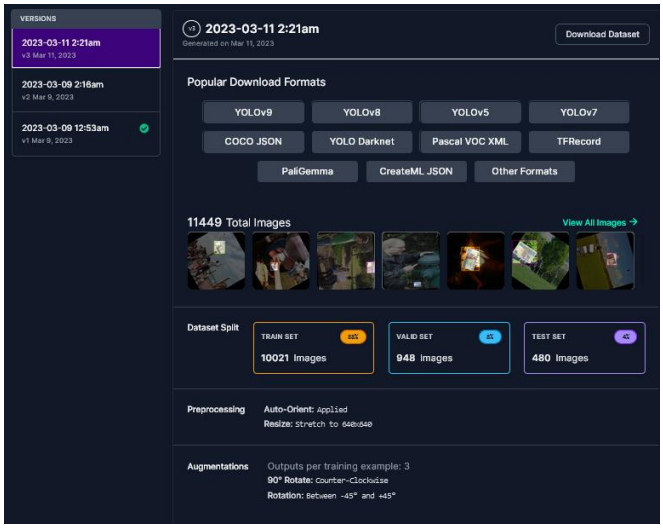
See This Gif ➔ Counts the Egyptian Currencies Detected!

3) **Text-to-Speech Feature:** Pressing the **'t'** key triggers a feature that Make a **Sound Feedback** that announces the total amount of money detected and lists the detected currencies from right to left **(relative to the camera)**

See this Video ➔ Sound Feedback (Feature)

3) Dataset

I Trained my model on the [Banha University Egyptian Currency Dataset] available on **Roboflow Universe.**



4) YoloV8 Model

YOLOv8 (**You Only Look Once**, Version 8) is the latest iteration in the YOLO series of real-time object detection models, known for **their efficiency** and **speed** ⚡ . YOLO models are designed to detect objects in images and videos by dividing the image into a grid and predicting bounding boxes and class probabilities for each grid cell simultaneously.

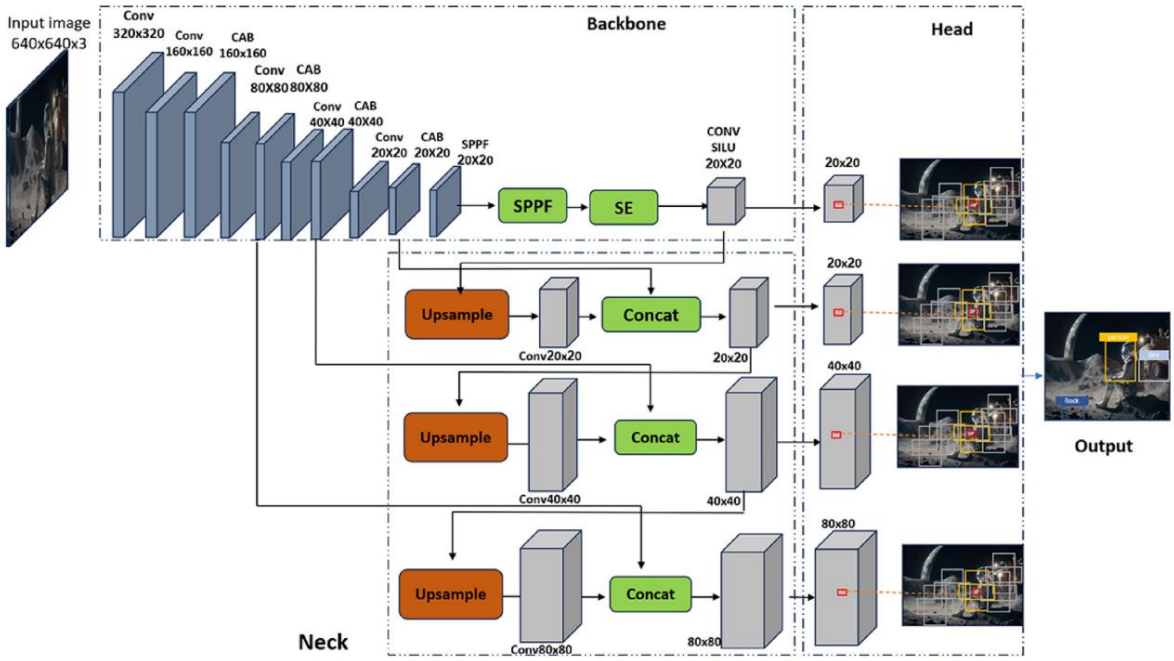


Fig. 1. Full Yolov8-CAB network architecture.

Architecture

YOLOv8's architecture is an evolution of previous YOLO models, utilizing a convolutional neural network divided into **two** main parts backbone and head: -

1) **The backbone:** is based on a modified version of the **CSPDarknet53** architecture, consisting of **53 convolutional layers** enhanced with **cross-stage partial connections**.

2) **The neck:** is designed to enhance the feature maps obtained from the backbone. It usually consists of layers like the Spatial Pyramid Pooling (SPP) and Path Aggregation Network (PAN).

   SPP helps by pooling features at different scales, while PAN improves the information flow by connecting the feature maps of different scales, aiding in the detection of objects at various sizes.

3) **The head:** Comprises **multiple convolutional layers** followed by fully connected layers responsible for predicting bounding boxes, objectness scores, and class probabilities.

Notably, YOLOv8 integrates a self-attention mechanism in the head of the network and a feature pyramid network for multi-scale object detection, enabling it to focus on various parts of an image and detect objects of **different sizes** and **scales**.

Limitations

Users should be cognizant of certain constraints when using the YOLOv8 model. It can face challenges in accurately identifying objects in environments with **significant clutter** or when objects are **only partly visible**. The model might also find it difficult to recognize smaller objects or those that don't stand out much against their **background**.

---

*And That's why I added **Gaussian Blur** to all the perceived images that goes to the model to only focus on the important features so that the model doesn't get Confused.*

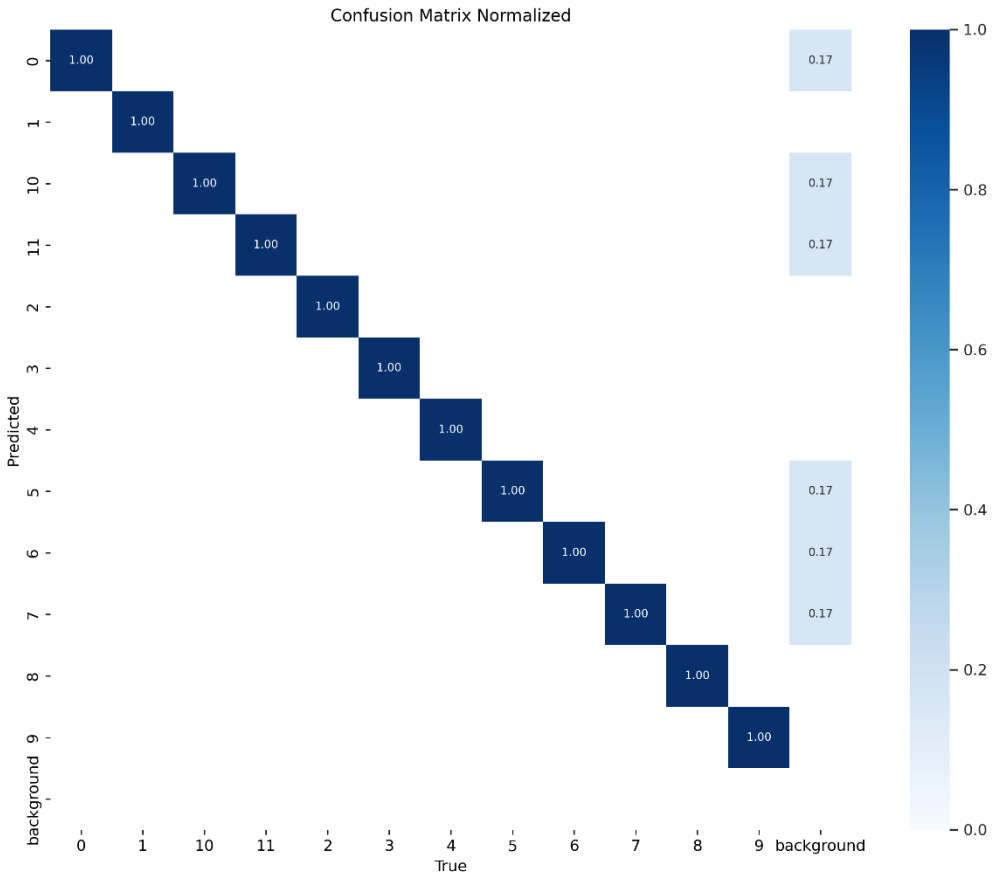
➔ To Know more about YoloV8, see this Paper: YoloV8 Research Paper

➔ And for more user-friendly information: Yolov8 user model Guide

5) My Results

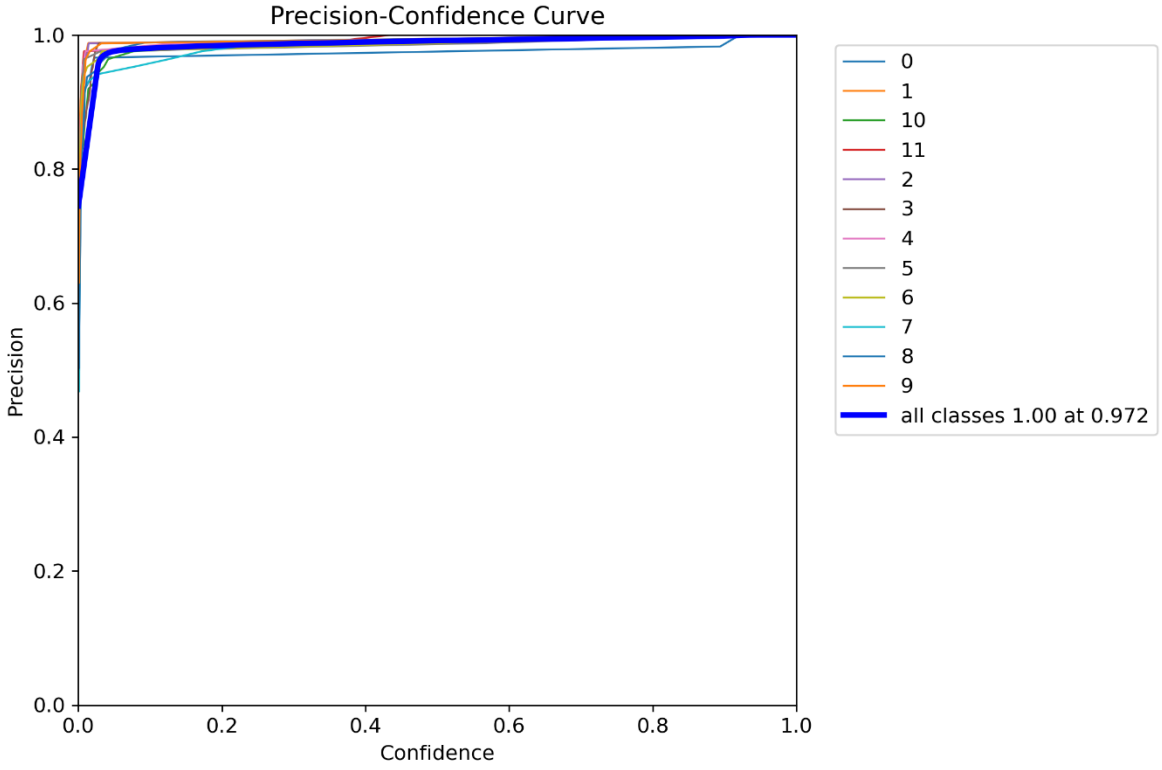
Confusion Matrix

- I got a very good Confusion matrix thanks to yolov8 model capabilities.
- But I noticed that sometimes it gets confused about the background (classifying it to some currency)



Precision Curve

In Precision Curve I got an Accuracy of **97.2%**



Epochs Metrics: -

Epochs	train/box_loss	train/cls_	train/dfl_	metrics/precision	metrics/recall(B)	metrics/mAP50	metrics/mAP50-95(	val/box_loss	val/cls_los	val/dfl_loss
1	0.58877	2.3182	1.0303	0.85784	0.80949	0.90641	0.8396	0.3663	0.73485	0.79543
2	0.55434	0.96298	0.98257	0.88807	0.83985	0.92959	0.87885	0.33299	0.60804	0.78902
3	0.54093	0.8155	0.97292	0.92136	0.92164	0.96837	0.92191	0.32884	0.45411	0.79313
4	0.51851	0.70353	0.96485	0.97595	0.97573	0.99191	0.92776	0.38369	0.32593	0.79711
5	0.4875	0.59914	0.94884	0.98715	0.99294	0.99342	0.93064	0.37002	0.28102	0.79176
6	0.3368	0.39016	0.8552	0.98203	0.99134	0.99415	0.94784	0.35028	0.26429	0.78721
7	0.31365	0.34334	0.84537	0.98801	0.98216	0.99398	0.96423	0.28918	0.24975	0.78198
8	0.30033	0.31589	0.83782	0.99251	0.99282	0.99474	0.97306	0.26809	0.22486	0.7762
9	0.28355	0.28178	0.83319	0.98411	0.98396	0.9888	0.97389	0.22814	0.2301	0.7713
10	0.26884	0.25963	0.82592	0.99495	0.99946	0.995	0.9752	0.26634	0.1997	0.77448
11	0.257	0.23796	0.82612	0.99718	0.99757	0.99499	0.97824	0.23068	0.18037	0.76815
12	0.24352	0.21693	0.82232	0.99535	0.99746	0.99499	0.98383	0.22109	0.17612	0.76502
13	0.2289	0.19818	0.81378	0.99826	0.99926	0.995	0.98276	0.23084	0.16363	0.76538
14	0.21796	0.18528	0.81312	0.99618	0.99819	0.995	0.98848	0.18635	0.14667	0.7612
15	0.20545	0.16928	0.80873	**0.99737**	0.99894	0.995	0.98974	0.1862	0.1418	0.76135

6) Conclusion

YOLOv8 is a continuation of the YOLO series by **Ultralytics**, focusing on real-time object detection with enhanced speed and accuracy. Its architecture is designed to efficiently process images, extract meaningful features, and provide precise object localization and classification, making it suitable for applications requiring rapid and reliable object detection.