**MODEL BUILDING**

This project explores using the YOLO (You Only Look Once) model for spotting Mounds in images. YOLO is famous for its ability to quickly detect objects in real-time. Our goal is to make the most out of YOLO's strengths to create a reliable system for identifying objects in various situations. By tapping into YOLO's smart design and advanced features, we aim to build a system that can accurately. We'll be looking at how well YOLO performs, how we can make it better, and where we can use it in the real world.

But before using YOLO there are certain tasks to be performed which involves

**Data Preprocessing**

Streamlining the process of preparing image data and annotations for training object detection models, specifically tailored for YOLO (You Only Look Once) format. automating the tasks such as parsing XML files, data preprocessing, feature engineering, data splitting, label encoding, and saving data in a standardized format.

**Model Training**

The YOLOv5 model is trained using a dataset specified in data.yaml for 50 epochs with a batch size of 8. The configuration file yolov5s.yaml is used to define the architecture of the YOLOv5 model. The training process involves optimizing the model using stochastic gradient descent (SGD) with a learning rate of 0.01. The training dataset consists of 109 images for training and 27 images for validation.

During training, various image augmentations are applied, including blur, median blur, grayscale conversion, and Contrast Limited Adaptive Histogram Equalization (CLAHE). These augmentations aim to enhance the model's ability to generalize to different conditions.The training progress is logged, including GPU memory usage, box loss, objectness loss, and class loss, along with the number of instances and image size for each epoch. Additionally, precision (P), recall (R), and mean Average Precision (mAP) metrics are calculated for evaluating the model's performance on both training and validation datasets.

**Prediction on Test Dataset**

We use YOLO\_Pred class from the yolo\_predictions module to make predictions on images using a pre-trained YOLOv5 model. The model weights are loaded from the file best.onnx, and the dataset configuration is specified in data.yaml.A folder containing test images containing unseen data is specified, and the code iterates through each image file in the folder. For each image, it loads the image using OpenCV (cv2). Then, it makes predictions on the image using the YOLO\_Pred object and displays the predicted image.Which produced the below results in figure(to be specified)

**Results**

**Train Box Loss:**

The train box loss metric measures the difference between the predicted bounding boxes and the actual bounding boxes of the objects in the training data. A lower box loss means that the model's predicted bounding boxes more closely align with the actual bounding boxes.

A graph with numbers and lines

Description automatically generated

**Validation Box Loss:**

The Validation box loss metric measures the difference between the predicted bounding boxes and the actual bounding boxes of the objects in the Validation data. A lower box loss means that the model's predicted bounding boxes more closely align with the actual bounding boxes.

A graph of a graph

Description automatically generated with medium confidence

**Train and validation Class Loss:**

The train class loss metric measures the difference between the predicted class probabilities and the actual class labels of the objects in the data. A lower class loss means that the model's predicted class probabilities more closely align with the actual class labels.

A graph with numbers and a line

Description automatically generated A graph of a line with numbers

Description automatically generated

**Evaluation Metrics:**

**Metrics Precision :**

The metrics precision metric measures the proportion of true positive detections among all the predicted bounding boxes. A higher precision means that the model is better at correctly identifying true positive detections and minimizing false positives.

A graph with blue and orange lines

Description automatically generated

**Metrics Recall :**

The metrics recall metric measures the proportion of true positive detections among all the actual bounding boxes. A higher recall means that the model is better at correctly identifying all true positive detections and minimizing false negatives.

A graph with blue and orange lines

Description automatically generated

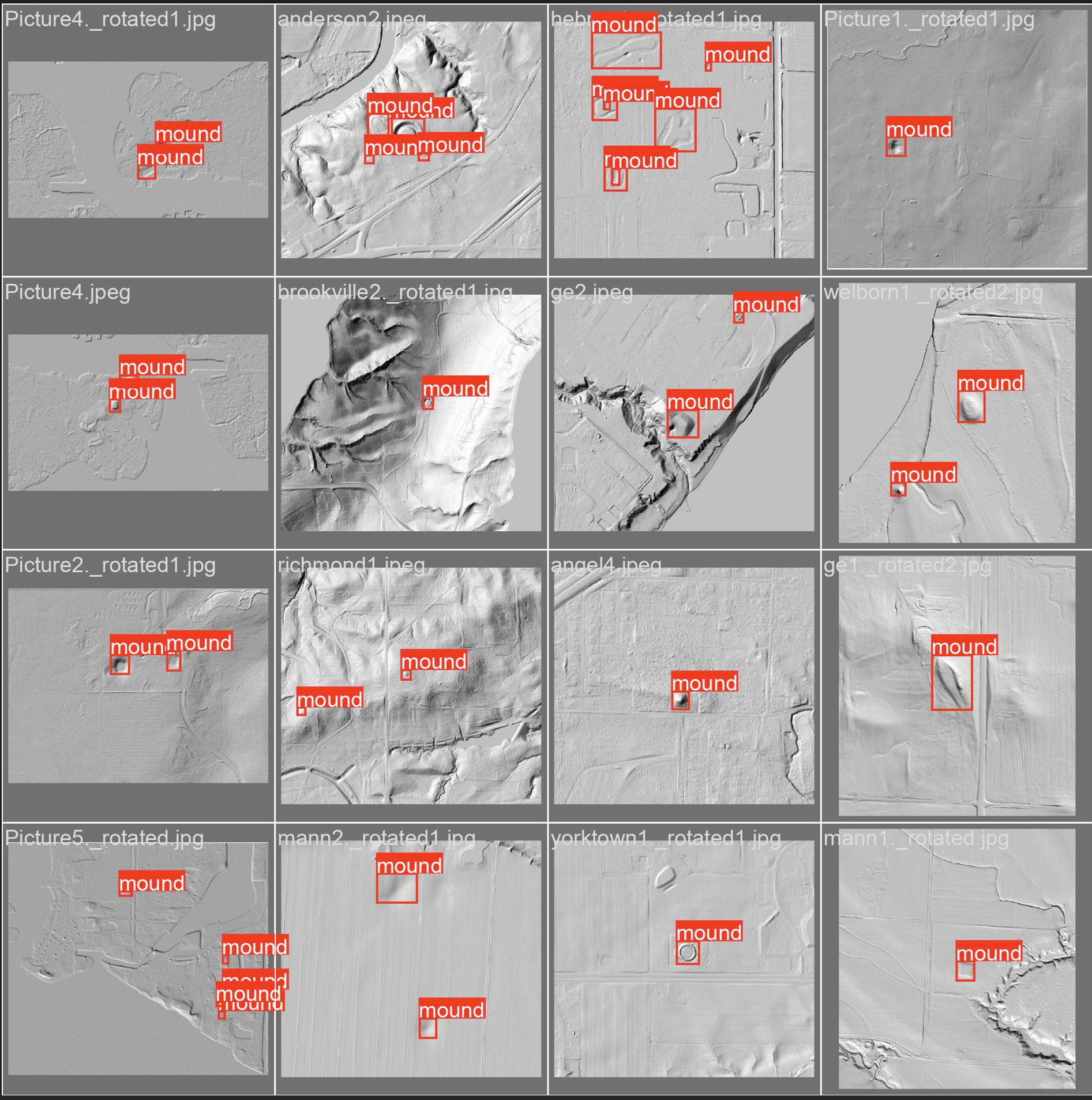
**Metrics mAP50 :**

The metrics mAP50 metric measures the mean average precision of the model across different object categories, with a 50% intersection-over-union (IoU) threshold. A higher mAP50 means that the model is better at accurately detecting and localizing objects across different categories.

A graph showing the growth of a number of points

Description automatically generated with medium confidence

**Validation images**

 A screenshot of a computer

Description automatically generated

To conclude, The precision score of 0.573 indicates that approximately 57.3% of the objects detected by the model were correct out of all the objects it predicted. Conversely, the recall score of 0.559 suggests that the model successfully identified approximately 55.9% of all ground truth objects present in the dataset. The similarity in scores between precision and recall indicates a balanced performance in both the accuracy of detections and the comprehensiveness of object detection.

**CONCLUSION**

We leveraged the Machine Learning Model YOLO for the identification of archaeological mounds using Digital Terrain Model (DTM) data. While previous research work uses Digital Elevation Model (DEM) and Machine Learning techniques like CNN and PointConv, we used Digital Terrain Model, which is a significant improvement. To overcome the limitations of CNN, we employed YOLO, which is better at object detection. Since this is an area of interest with ongoing research, we wish to continue by improving the dataset quantity through the implementation of GANs, enhancing the precision and accuracy of the model, and deploying it to make predictions on the entire state of Indiana. Additionally, we aim to generate a probability map that provides the confidence level of the predictions made at every location in the state of Indiana.