Department of Electrical and Computer Engineering
North South University

Autonomous Volcanic Terrain Exploration System

Mahadi Melon Soykat ID #1921950042

Shaznin Jahan Sudha- 1721636042

Md Tahsinul Islam-1921232042

Faculty Advisor:

Dr. Mohammad Shifat-E-Rabbi

Assistant Professor

ECE Department

Spring, 2024

ABSTRACT

Autonomous Volcanic Terrain Exploration System

Abstract: In order to explore volcanic terrains effectively and accurately, this research introduces a "Autonomous Volcanic Terrain Exploration System," which makes use of YOLOv11's sophisticated instance segmentation capabilities. The system uses a custom-trained dataset to recognize, categorize, and segment objects and geological characteristics in difficult situations. By putting YOLOv11 into practice, the initiative hopes to lower hazards to human researchers by enhancing autonomous navigation and data collecting in remote and dangerous volcanic locations. The development process is described in this article, along with the system's possible uses in robotics and geoscience, from dataset preparation to model training and deployment.

I. Introduction:

Because of their severe and unpredictable conditions—such as high temperatures, toxic gasses, and unstable ground—exploring volcanic terrains is extremely difficult. Because of the significant risks these environments present to human researchers, autonomous devices that can navigate and gather data efficiently are essential. By utilizing developments in computer vision and deep learning, the "Autonomous Volcanic Terrain Exploration System" seeks to overcome this difficulty.

The core of this system is YOLOv11, a state-of-the-art instance segmentation algorithm renowned for its great accuracy and real-time processing speed. YOLOv11 can recognize and segment important geological features and possible hazards on its own by being trained on a unique dataset designed for volcanic environments. This feature improves the quality of data for scientific inquiry while also enabling safer exploration.

The project's goals, design process, dataset preparation, model training, and possible uses are all covered in detail in this study, which also highlights how AI-driven solutions might advance the fields of environmental monitoring and autonomous exploration.

II. Research Literature Review:

A. **Towards energy efficient autonomous exploration of Mars lava tube with a Martian coaxial quadrotor:** This paper presents an autonomous exploration mission for a Martian lava tube using a modified frontier approach and a risk-aware planning and integrated collision avoidance scheme. The system is designed to be energy efficient and to be able to explore complex and challenging environments. The paper presents a simulation study of the system in a Martian lava tube environment. [1]

B. **Proximal Exploration of Venus Volcanism with Teams of Autonomous Buoyancy-Controlled Balloons:** This paper presents a concept for exploring Venus's atmosphere using a team of autonomous buoyancy-controlled balloons. The balloons would be equipped with a variety of sensors to study Venus's atmosphere and surface. The paper presents a simulation study of the system in Venus's atmosphere. [2]

C. **Volcanological applications of unoccupied aircraft systems (UAS): Developments, strategies, and future challenges:** This paper provides an overview of the use of unmanned aerial systems (UAS) for volcanic research. It discusses the advantages and disadvantages of using UAS for volcanic studies, as well as the challenges that need to be overcome. The paper also highlights some of the most promising applications of UAS for volcanic research, such as monitoring volcanic activity, mapping volcanic terrain, and sampling volcanic gases. [3]

III. Methodology:

A. Dataset Creation:

One of the most important aspects of the Autonomous Volcanic Terrain Exploration System was building a strong and varied dataset. The objective was to create a labeled dataset with pictures of volcanic landscapes marked with dangers including fire, gas, and lava. The steps involved in creating a dataset are as follows:

i. Data Collection:

Twenty high-resolution photos of volcanic settings with lava flows, gas emissions, and fire plainly visible were collected in order to gather data. These photos are from licensed media channels, research publications, and public repositories. A strong basis for model training was provided by the carefully chosen variety of scenarios, which included various terrains, lighting conditions, and eruption stages.

B. Creation Process:

To guarantee precise and thorough annotations that would properly train the YOLOv11m-seg model, a well-defined process was used to create the dataset for the Autonomous Volcanic Terrain Exploration System. We did this by categorizing 20 high-resolution photos of volcanic landscapes that showed lava flows, gas emissions, and fire in plain sight using Label Studio, an open-source data labeling tool. Label Studio was chosen because it is user-friendly, supports a wide range of annotation types, and can manage enormous amounts of data—even when high accuracy is needed for tasks like object detection and segmentation.

i. Uploading Images to Label Studio:

Uploading the 20 chosen photos into Label Studio was the first stage in the procedure. Lava, gas, and fire are the three main threats that are clearly depicted in these pictures, which were selected to represent various volcanic circumstances. To guarantee the accuracy of the annotations, every image was uploaded into the platform in its original high-resolution format. After uploading, the pictures showed up in the workspace, prepared for annotation. As a result, we were able to start the meticulous process of getting the data ready for the subsequent phases of model training.

ii. Label Creation:

The next step was to specify the precise labels we wanted the model to recognize after the photos were uploaded. These labels matched the volcanic dangers depicted in the pictures:

- LAVA: This term was applied to any flowing lava or molten rock that could be seen in the pictures. Lava was frequently encircled by smoke or ash and was distinguished by its intense orange or red glow.
- GAS: The gas designation was used for regions where visible emissions of smoke, steam, or harmful gasses came from earth fissures or volcanic eruptions. The shape and density of gas had to be carefully considered because it was frequently more diffuse than lava or fire.
- FIRE: The fire label was used for any visible flames or burning material in the volcanic environment. This included both wildfires caused by volcanic activity and the fiery nature of molten lava interacting with surrounding vegetation or structures.

iii. Choosing Colors for Boundaries

The visualization of the discovered items' borders is a crucial component of object detection and segmentation. To make it easier to recognize the labels during the annotation and training process, users can use Label Studio to apply various colors to their boundaries. In order to differentiate the visual representation of lava, gas, and fire for this project, we selected particular colors for each term.

iv. Annotation of the Images

The actual annotation of the photographs came next, after the labels and boundary colors were set. The most intricate and time-consuming step in the dataset creation process is this one. Every image was examined by hand, and using Label Studio's annotation tools, the regions that contained lava, gas, or fire were meticulously delineated.

v. Processing and Dataset Download

Following the annotation of every image, we moved on to Label Studio's processing stage. In this step, the annotations were finalized and formatted so that the YOLOv11 model could be trained. Annotated data may be exported from Label Studio in a number of forms, including JSON, which was used for this project since it works with YOLO-based models.

The dataset with annotations was available for download. This step was essential since it gave us the final dataset in a structured format that we could use straight into the YOLOv11 training pipeline. All of the annotations, including the segmentation masks and bounding boxes for lava, gas, and fire, as well as metadata on the image dimensions and class labels, were included in the downloaded dataset.

C. Data Processing

A comprehensive data pretreatment pipeline was developed to get the dataset ready for use with YOLOv11 in order to guarantee compatibility and peak performance during model training. The YOLOv11 model was chosen because it strikes a compromise between speed, accuracy, and computational efficiency. This is especially true of its segmentation form, YOLO11m-seg. YOLO11m-seg has the following features: 640 pixels in size, 51.5 mAPbox (50-95) and 41.5 mAPmask (50-95), 281.6 ± 1.2 CPU ONNX (ms), 6.3 ± 0.1 T4 TensorRT10 (ms), 22.4 parameters (M), and 123.3 FLOPs (B). These measures demonstrated how well the model handled segmentation tasks, giving it a great option for identifying and classifying volcanic threats like fire, gas, and lava.

i. Model Preparation and File Download

The official YOLOv11 website provided the YOLO11m-seg model file (YOLO11m-seg.pt). The pre-trained model weights, which were the foundation for fine-tuning on our unique dataset, were included in this file. Reduced training time and processing resources were needed because the pre-trained model, which was optimized for segmentation tasks, was made to function effectively on datasets comparable to ours. During the training process, the file was easily accessible because it was kept in the project directory.

ii. Dataset Division

The dataset was split into two sections to guarantee accurate model evaluation:

Training Set: The model was trained using this subset of the dataset, which enabled it to identify trends and characteristics associated with volcanic hazards. The training set included about 75% of the total photos.

Validation Set: 15% of the dataset was set aside just for validation. During training, this set was used to track the model's performance to make sure it wasn't overfitting and could effectively generalize to new data.

iii. Creating the Dataset Configuration File

A configuration file called dataset_custom.yaml was made in order to train YOLOv11 on the custom dataset. The model received crucial dataset information from this YAML file, such as:

- Paths: To make sure the model could find the pictures and annotations during training, the file included the paths to the training and validation datasets.
- Number of Classes: It was stated how many labels there were in the dataset (in our example, three: lava, gas, and fire).
- Class Names: The model was able to link predictions to particular dangers since the class names were stated clearly (e.g., "LAVA," "GAS," and "FIRE").

iv. Training Script Setup

The training procedure was then managed by a Python script called train.py. This script defined the model's training parameters and made use of the YOLOv11 framework. Among the script's crucial steps were:

Model Loading: The YOLO11m-seg model file (YOLO11m-seg.pt) was loaded by the script as the training base model.

Dataset Integration: To load the training and validation datasets, the script made reference to the dataset_custom.yaml file.

Configuration of Hyperparameters:

- Batch Size: The number of images processed in a single iteration which was set 8.
- Epochs: The number of times the entire dataset would be passed through the model during training which was set 100.
- Learning Rate: The rate at which the model adjusted its weights.
- Image Size: The resolution (640 pixels) at which the images would be processed.

IV. Training the Model

With the script prepared, the train.py file was launched through the Anaconda Prompt, a popular command-line interface for Python-based applications, to begin the training process. To guarantee compatibility, the necessary Python packages (such as PyTorch and the YOLOv11 library) were installed in the Anaconda environment prior to the script's execution.

The model loaded the pre-trained weights from YOLO11m-seg.pt and adjusted them on the custom dataset to start the training process. In order to minimize the loss function—which quantified the discrepancy between the model's predictions and the actual annotations—the weights were iteratively modified once the training set's photos and annotations were fed into the model.

Metrics like mAP (mean Average Precision) for masks (mAPmask) and bounding boxes (mAPbox) were computed during training using the validation set. The effectiveness of the model's learning to identify and categorize dangers was revealed by these metrics. By routinely tracking these data, possible problems like overfitting or underfitting may be found and the training parameters could be changed.

After Training Result:

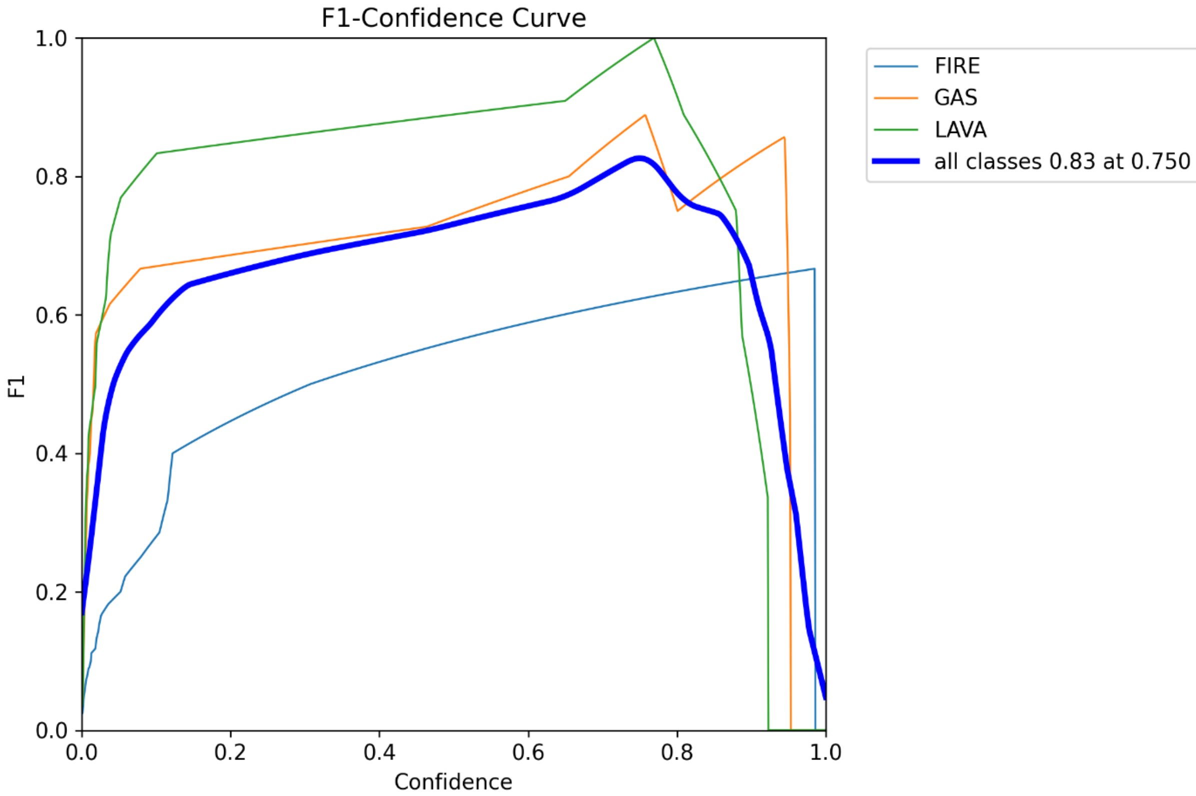

Fig 1: F1 Confidence Curve

For each class (FIRE, GAS, and LAVA), the F1-Confidence Curve illustrates the correlation between the model's F1 score and prediction confidence. At a specific confidence level, the curve for "all classes" shows the average F1 score for every class. The model reaches its maximum overall F1 score (0.83) at a confidence threshold of 0.75, according to the "all classes" curve. This indicates that the model is probably accurate if it predicts a class with a confidence level of 0.75 or above.

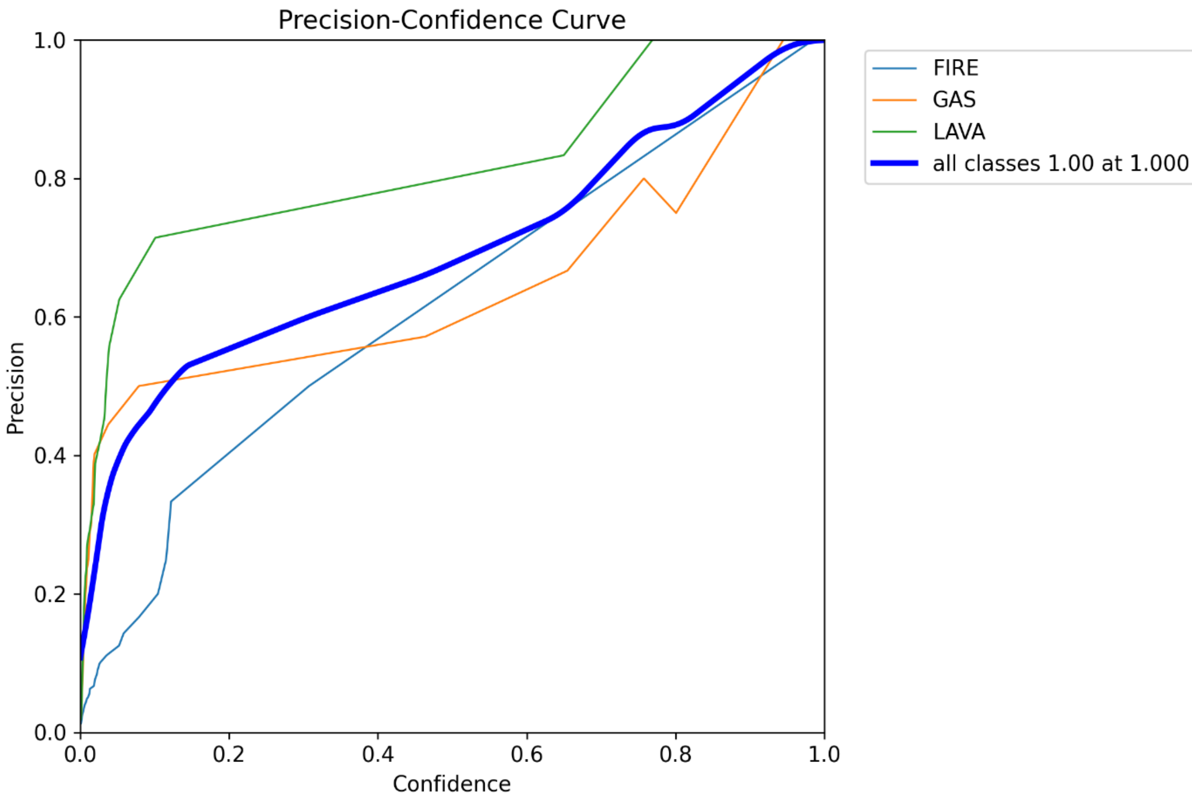

Fig 2: Precision-Confidence Curve

For each class (FIRE, GAS, and LAVA), the Precision-Confidence Curve illustrates the correlation between the model's precision and forecast confidence. The average precision for every class at a specific confidence level is shown by the "all classes" curve. The model reaches its maximum overall precision (1.00) at a confidence level of 1.000, according to the "all classes" curve. This implies that the model is assured to be accurate if it predicts a class with a confidence level of 1.00.

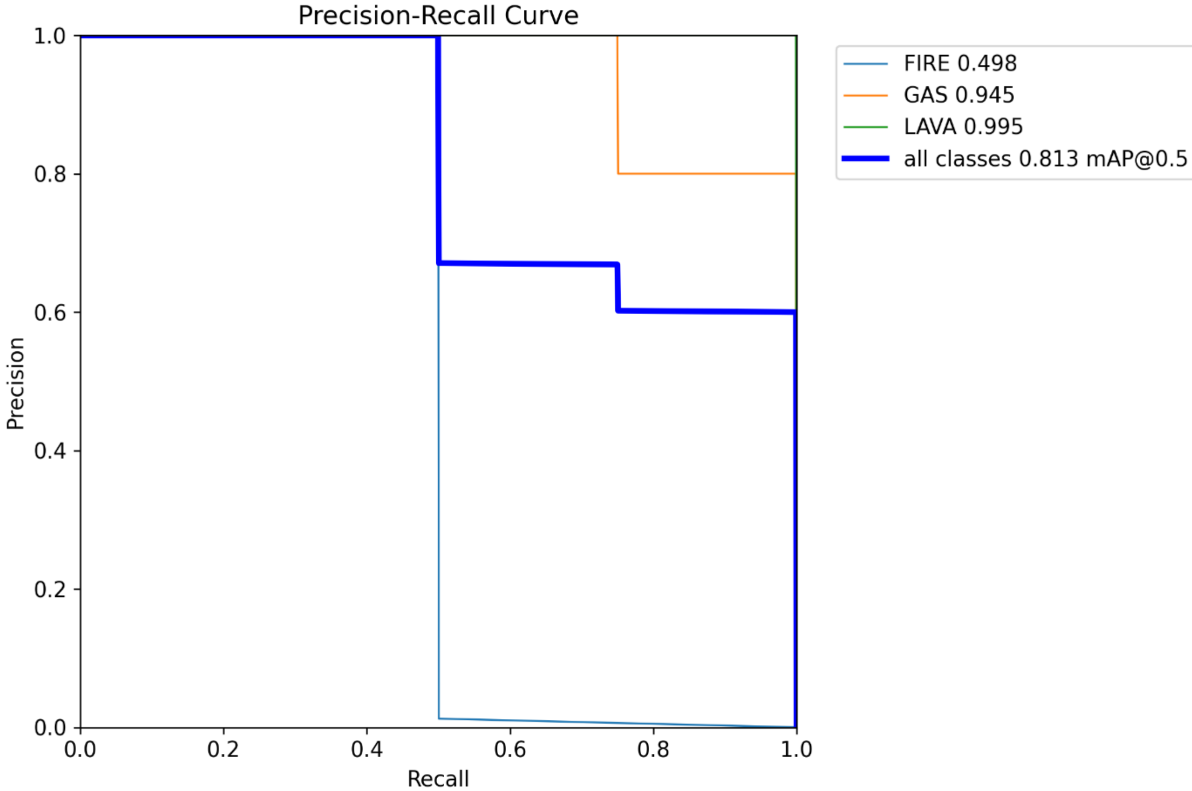

Fig 3: precision and recall Curve

The link between precision and recall for several classes (FIRE, GAS, and LAVA) is displayed by the Precision-Recall Curve. The average precision-recall curve for all classes is represented by the "all classes" curve.The model achieves a mean Average Precision (mAP@0.5) of 0.813, according to the "all classes" curve. Although this result is reasonable, it may be better, particularly for the "FIRE" class.

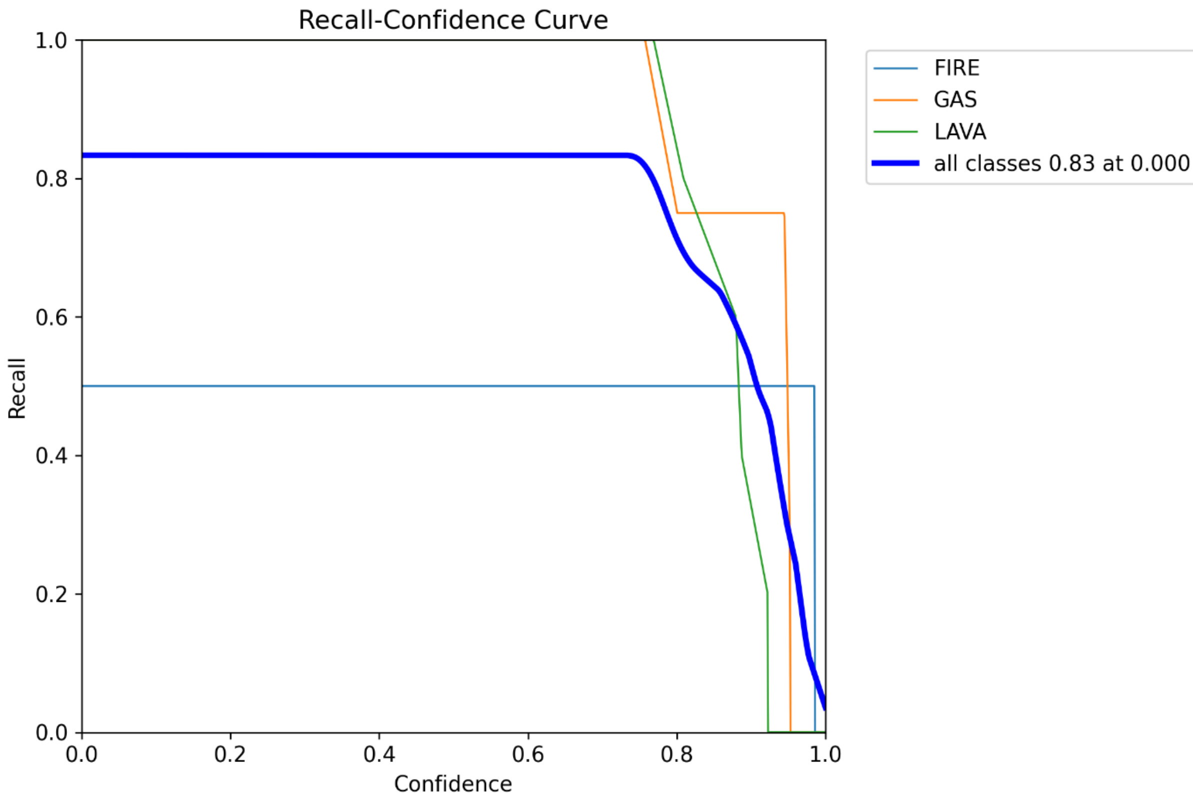

Fig4: Recall-Confidence Curve

For each class (FIRE, GAS, and LAVA), the Recall-Confidence Curve illustrates the correlation between the model's recall and prediction confidence. The average recall for every class at a specific confidence level is shown by the "all classes" curve.The model reaches its maximum overall recall (0.83) at a confidence level of 0.000, according to the "all classes" curve. This indicates that even at extremely low confidence levels, the model can detect a sizable percentage of real positive cases. At low confidence levels, this also raises the possibility that the model has a high false positive rate.

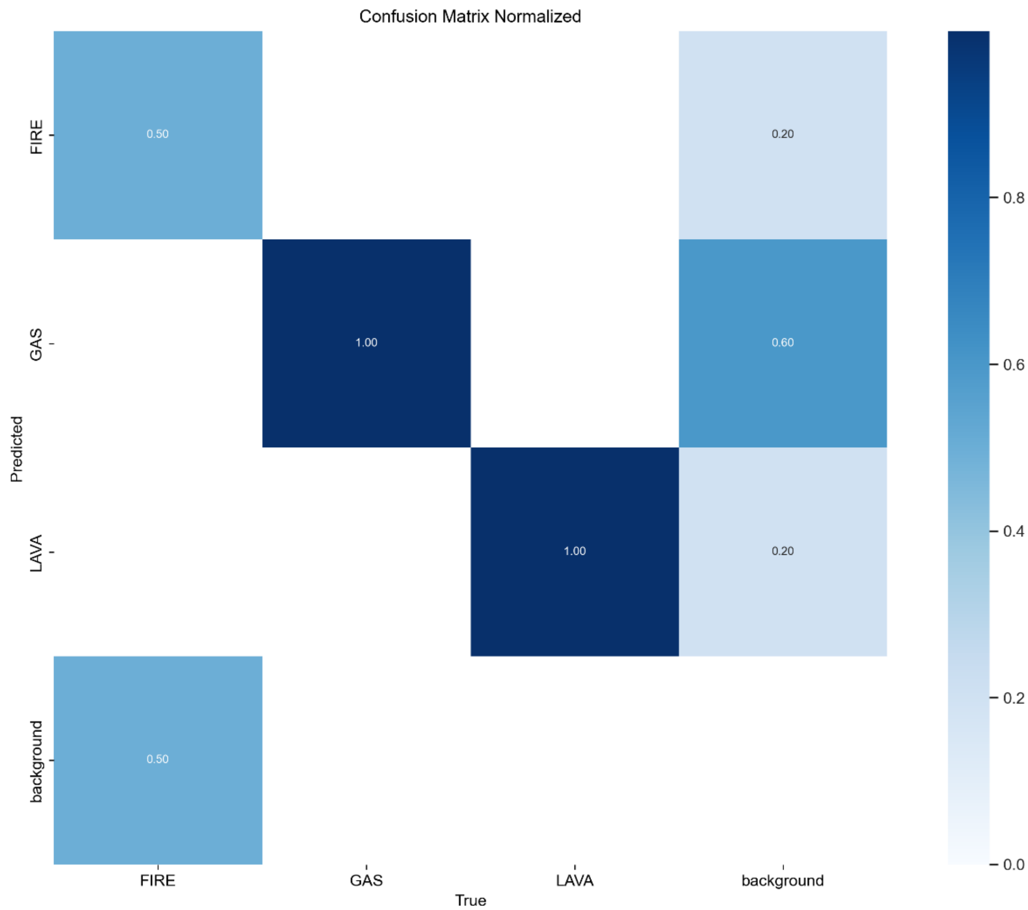

Fig 5: Confusion Matrix Normalized

A normalized representation of the model's predictions can be seen in the given confusion matrix. Correct classifications are represented by diagonal elements, and incorrect classifications are shown by off-diagonal elements. The model appears to function rather well based on the facts provided. Its classification accuracy for the "LAVA" and "GAS" categories is high. As can be seen by the off-diagonal numbers, it has trouble with the "FIRE" and "background" categories.

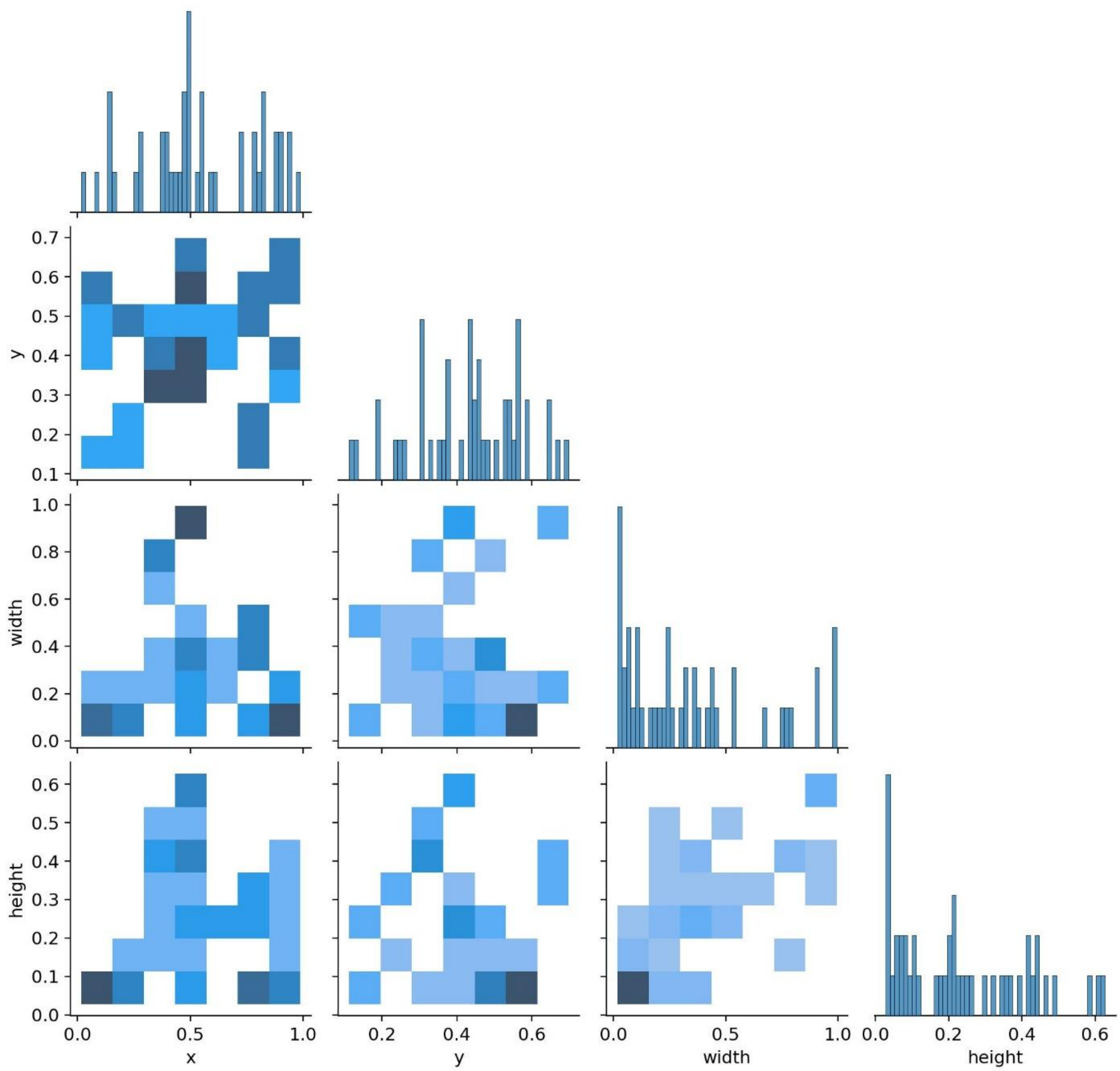

Fig 6: Label Plots

The given figure, which shows the relationships between four variables (x, y, width, and height), looks to be a pair plot. A scatterplot or histogram for a pair of variables is displayed in each subplot.

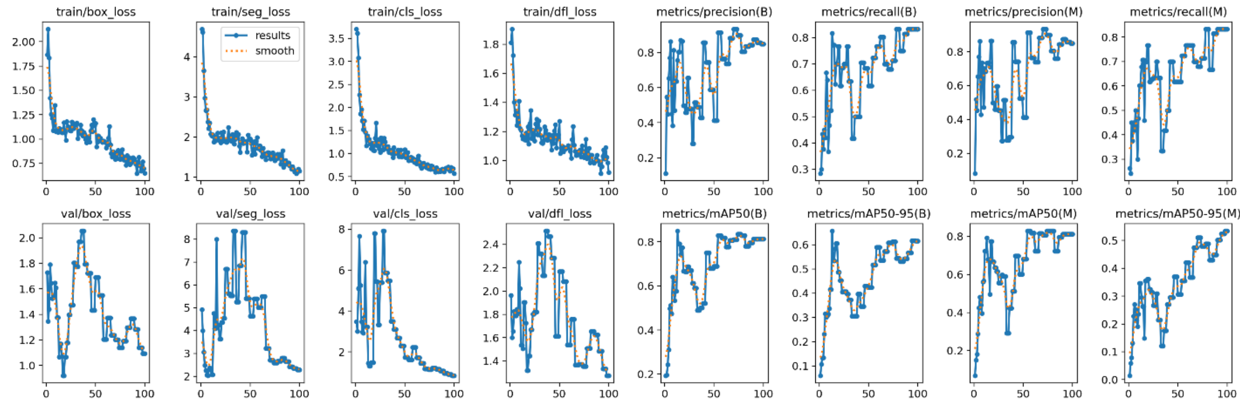

Fig 7: training and validation curves

The training and validation curves for different loss functions and metrics during machine learning model training are displayed in the graphic that is provided.

V. Deployment:

Deploying the model for real-time detection was a critical step after it had been successfully trained. During this stage, a script that would use the learned model for inference on fresh images or video streams had to be set up. This was accomplished by writing a Python script called predict.py to manage the deployment procedure.

A. Linking the Trained Model

Loading the trained model was the initial step in predict.py. The model file, best.pt, which included the optimized weights following training, was connected for this reason. Using the learnt patterns from the unique dataset, this model was able to identify and categorize the three volcanic hazards: fire, gas, and lava. We utilized the YOLOv11 framework to load the model, which offered the functions required to load the.pt file and get it ready for inference.

B. Declaring the Source for Inference

Declaring the source from which the input photos or video will be taken came next after loading the model. To instruct the model to handle an image or video from a certain file directory, the source was first set to that path in predict.py. However, we included the ability to dynamically switch the source to provide deployment flexibility. By changing the source to 0 can access the webcam.

VI. Final Result:

Fig 8: Result Demo

By running the model one a JPG file we were able to detect LAVA, GAS and Fire with there confidence score.

VII. Conclusion:

An important step toward increasing safety and effectiveness in the detection of volcanic dangers like lava, gas, and fire is the Autonomous Volcanic Terrain Exploration System employing YOLOv11. This effort effectively created a strong model that can reliably identify these risks in real-time from both static photos and live video streams by utilizing deep learning.

The system was trained to identify and segment volcanic hazards with high precision by applying performance measurements from YOLO11m-seg and carefully creating datasets using Label Studio. The model's potential application in real-world settings, like robotic exploration systems or volcanic monitoring stations, where prompt danger detection can be crucial, is made possible by its capacity to identify lava, gas, and fire from photos and videos.

The success of this project depended heavily on the phases of data preprocessing, model training, and deployment. The system achieved remarkable performance metrics through the usage of YOLOv11m-seg and careful consideration of data augmentation, validation, and model optimization. Real-time inference was made possible during the deployment phase by the development of the predict.py script, which offers beneficial applications for instantaneous hazard detection. This is especially helpful in settings where volcanic activity is a serious danger.

In the end, this study shows how cutting-edge computer vision methods may be combined with practical uses like exploring volcanic terrain. This method not only improves safety but also paves the way for more independent and effective volcano monitoring and exploration technologies by automating the detection of hazardous materials including lava, gas, and fire.

References

[1] Akash Patel Samuel Karlsson Bjo¨rn Lindqvist, Christoforos Kanellakis ,Ali-Akbar Agha-Mohammadi , George Nikolakopoulos , "Towards energy efficient autonomous exploration of Mars lava tube with a Martian coaxial quadrotor," 11 2020. [Online]. Available: https://www.illumineation-h2020.eu/fileadmin/shares/illuMINEation/Publications/Towards_energy_efficient_autonomous_exploration_of_Mars_lava.pdf.

[2] Federico Rossia, Ma´ıra Saboiaa, Siddharth Krishnamoorthya, Joshua Vander Hooka, "Proximal Exploration of Venus Volcanism with Teams of Autonomous Buoyancy-Controlled Balloons," [Online]. Available: https://www.federico.io/pdf/Rossi.Saboia.Krishnamoorthy.Hook.ea.Acta23.pdf.

[3] M. R. James, "Volcanological applications of unoccupied aircraft systems (UAS): Developments, strategies, and future challenges," Vols. Volcanica, vol. 3, no. 1, pp. pp. 67–114, , Apr. 2020.