Using Sage Nodes for Edge AI in Fire Detection and Real-Time Data Monitoring

Giorgio Tran

Laboratory for Advanced Visualization & Applications,
Information and Computer Sciences
University of Hawaii at Manoa
ttran2@hawaii.edu

Abstract

Wildfires, such as the 2023 Lahaina fires, highlight the urgent need for advanced detection and monitoring technologies to facilitate timely emergency responses. YOLOv7 was fine-tuned on the D-Fire dataset and deployed on GPU-enabled Sage nodes for real-time smoke and fire detection. The model achieved a test set mAP@0.5 of 0.783, effectively identifying fire and smoke, though improvements are needed to reduce false positives. A data visualizer app within the SAGE3 collaboration platform was developed to ensure the reliability of both Sage nodes and Mesonet sensors, enabling users to compare data from both systems for real-time monitoring. Starlink satellite internet was evaluated for connectivity, revealing both benefits and limitations. Despite routing inefficiencies and latency, Starlink offers critical connectivity when traditional cell networks fail, such as during infrastructure loss in disasters. The integration of Edge AI, satellite internet, and the data visualizer app demonstrates potential for enhancing disaster management systems, especially in remote or infrastructure-compromised areas.

1 Introduction

In August 2023, deadly wildfires spread across Lahaina on the island of Maui, impacting the lives of many and causing an estimated $5.5B of damage. Similarly, between 2019 and 2020, up to 19 million hectares were burnt in Australia, and nearly 3 billion animals were affected by the blazes, according to the World Wildlife Fund (WWF). Edge sensors equipped with GPUs, such as Sage nodes, can offer critical support by running AI models directly at the edge for disaster detection, enabling faster response times for emergency services.

This project has three objectives:

- deploy machine learning models at the edge
- determine potential limitations to Starlink's network for emergency response purposes
- develop a data monitoring and comparison platform for Sage nodes and Mesonet stations

1.1 Sage Nodes

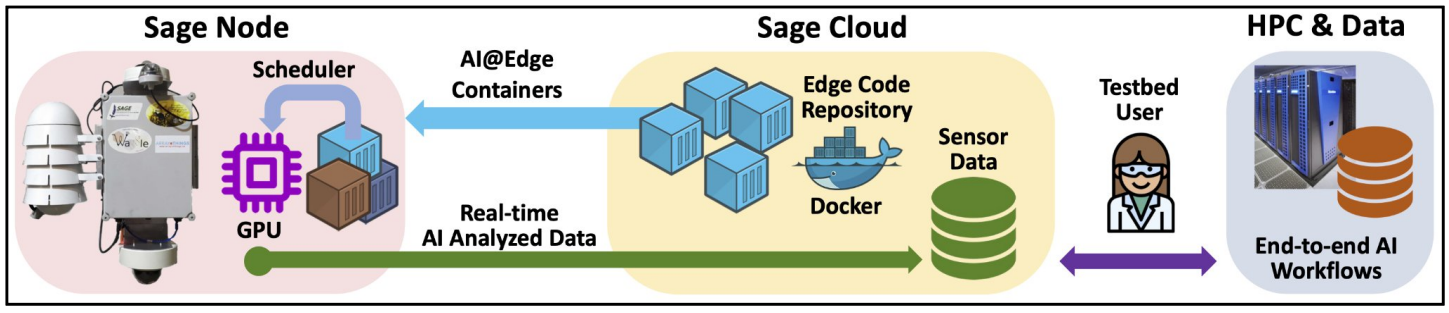

Figure 1: Sage node high level architecture. From: sagecontinuum.org

Sage Nodes (formerly known as Waggle) [1] are equipped with an NVIDIA Jetson Xavier NX compute module and a variety of environmental sensors, such as the Vaisala WXT and AQT series. In addition, Sage Nodes feature pan-tilt-zoom (PTZ) cameras for monitoring the surrounding environment. The data collected by these sensors is transmitted to the Beehive, a cloud-based infrastructure that allows users to access and interact with the data from the Sage Nodes.

1.2 Mesonet Stations

The Hawaii Mesonet aims to establish a climate monitoring system with approximately 100 stations to enhance the understanding of weather patterns across the Hawaiian islands and support weather and climate research. These stations collect environmental data similar to that of Sage Nodes, allowing for comparison between the two systems to ensure the accuracy and reliability of the data from both sources.

Figure 2: Sage node and Mesonet station on the same infrastructure in Lahaina. (A1-5): Sage node and its sensors. (A1): The Sage node with the BME680 sensor, top (XNF-8010RV) & bottom (XNV-8081Z) cameras, and raingauge. (A2): Mobotix infrared camera. (A3): Hanwha PTZ camera. (A4): Vaisala WXT536 weather sensor. (A5): Vaisala AQT530 particulate sensor. (B1-B4): Mesonet station and its sensors. (B1): Wind sensor. (B2-3): Temperature & humidity sensors. (B4): Enclosure for electronics used in data collection.

2 Methods

2.1 Fine Tuning and Deploying YOLOv7 to the Edge

The YOLOv7 [9] object detection model was fine tuned using the D-Fire dataset [8] to detect smoke and fire. This model was then deployed to the edge for real-time wildfire detection.

Figure 3: YOLOv7 fire and smoke detection from a training batch.

2.1.1 D-Fire Dataset

The D-Fire dataset contains a total of 21,527 images and labels: 17,221 images and labels are in the training directory, and 4,306 are in the testing directory. The labels are in a .txt file, formatted as follows:

<object-class> <x-center> <y-center> <width> <height>

In the test directory, 2,301 images have been labeled with bounding boxes, and 2,005 are negative samples without the bounding box. In the train directory, 9,388 images have been labeled with bounding boxes, and 7,833 images are negative samples. According to one of the main developers and authors for YOLO, Alexey Bochkovskiy (https://github.com/AlexeyAB), it is desirable that the training set includes images with non-labeled objects that we do not want to detect, and to use as many images of negative samples as there are images with objects. The D-Fire dataset adheres to those paradigms, where labeled and unlabeled data are split approximately 50/50. This helps the model

distinguish between objects of interest and the background or other unrelated objects more effectively.

2.1.2 Fine Tuning

The training ran for 43 epochs over the span of 12 hours with a batch size of 20 (which was the maximum batch size available for an NVIDIA P100 GPU) on Kaggle.

2.1.3 Deployment to the Node

An app running YOLOv7 was developed and tested on Sage node W097, located in Volcanoes National Park in Pahoa, Hawaii Island. Another Sage node was installed in Lahaina Sage, and a job was submitted to have the app running there as well. On the Sage node, the app operates within a Docker container, as illustrated in Figure 1, and accepts the following arguments: object confidence threshold, intersection-over-union (IOU) threshold, continuous operation mode, and the sampling interval between inferences.

2.2 Starlink Traceroutes and Bandwidth Tests

2.2.1 Traceroutes

Traceroutes were conducted on Starlink to analyze packet paths to various global endpoints, along with bandwidth performance tests. A traceroute works by sending a data packet toward a target with its Time-To-Live (TTL) initially set to 1. The first router decrements the TTL by 1, triggering a TTL-exceeded message that is sent back to the host. The host then increments the TTL by 1, allowing the packet to travel one hop further before the process repeats. This continues until the destination is reached or the maximum number of hops is exceeded. Once completed, the traceroute displays all the hops along with the time taken for each hop and its return.

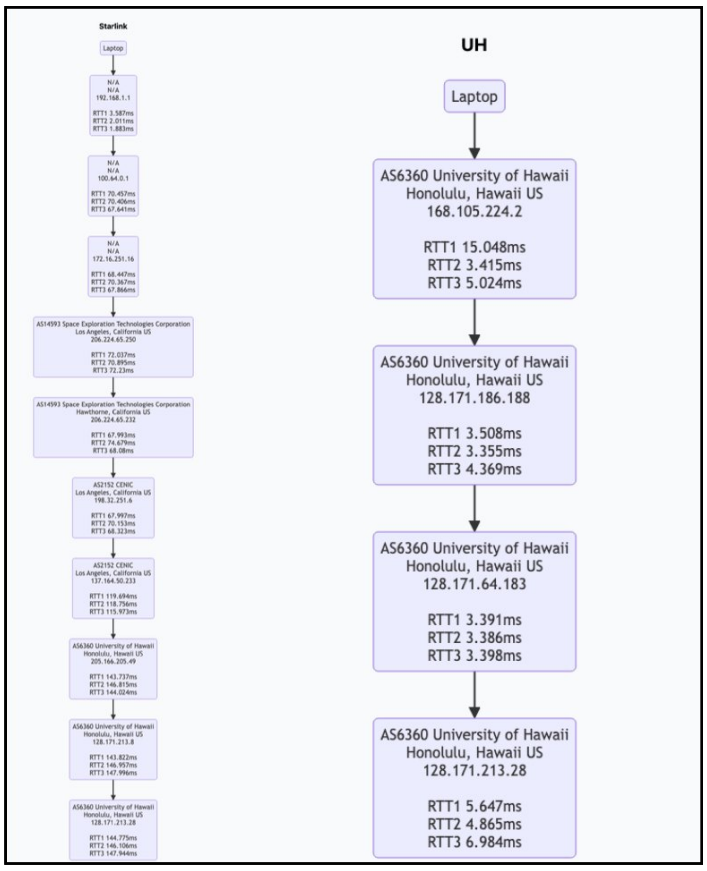

Figure 4: Traceroutes from Starlink to the University of Hawaii (UH) vs. UH to UH endpoints. Starlink routes go to California before returning to Hawaii.

12 endpoints were selected and they include the following locations: Hawaii, California, Thailand, Japan, Canada, Amsterdam, American Samoa, Guam, South Africa, South Korea, Northern India, and Southern India.

To determine the location of each hop, three IP databases (ip2location [3], ipgeolocation [4], and ipinfo [5]) were used to get an approximation.

A SAGE3 [7] plugin was developed to visualize the traceroutes for all of the countries. The visualizer uses Mermaid, a Javascript library, to show flow diagrams of each traceroute as shown in Figure 4.

2.2.2 Bandwidth Tests

The bandwidth tests were performed using both Ookla and the Starlink mobile application.

2.3 Data Visualizer

A data visualizer was developed on SAGE3 to enable users to compare data from Mesonet stations and Sage Nodes.

2.3.1 Data Processing

Certain measurements are shared between Mesonet stations and Sage Nodes, including rainfall, air temperature, relative humidity, wind speed, wind direction, and air pressure.

Each system has its own data hub, resulting in distinct APIs, variables, and data collection frequencies. Sage Nodes collect data at irregular intervals ranging from every second to every thirty seconds, whereas Mesonet stations collect data at a fixed interval of every 15 minutes. To align the Sage Nodes' data with the Mesonet stations' intervals, the Sage Nodes' data was interpolated to match as closely as possible within a window of ±1 minute.

2.3.2 User Interface

The user interface of the SAGE3 data visualizer consists of two main components: a map for selecting Mesonet stations (Figure 5) or Sage Nodes and their respective metrics, and the visualization (Figure 6). Users can select multiple stations per visualization to compare metrics across them. Since Sage Nodes and Mesonet stations share some metrics but also have unique ones, the interface adjusts accordingly: when both Sage Nodes and Mesonet stations are selected, only shared metrics are displayed. However, if only Sage Nodes or only Mesonet stations are selected, their unique metrics become available. For example, Sage Nodes include metrics like CPU and GPU utilization. This design enables seamless comparison both across and within sensor types.

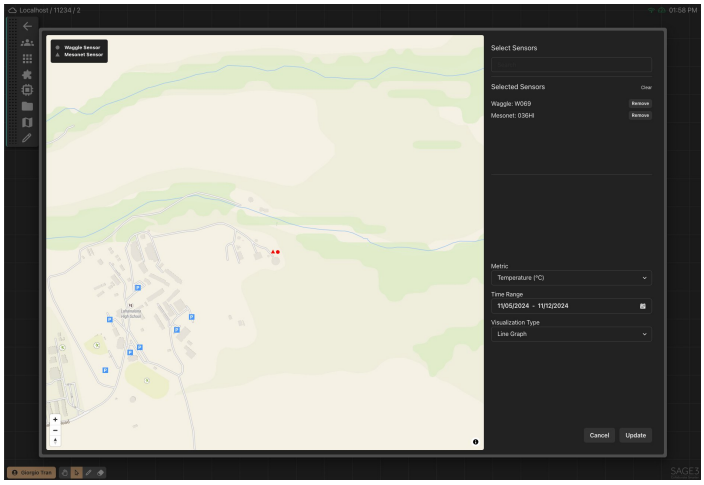

Figure 5: Station Selection Menu in SAGE3. Users can select stations to compare, choose the metrics to evaluate, and specify the type of visualization.

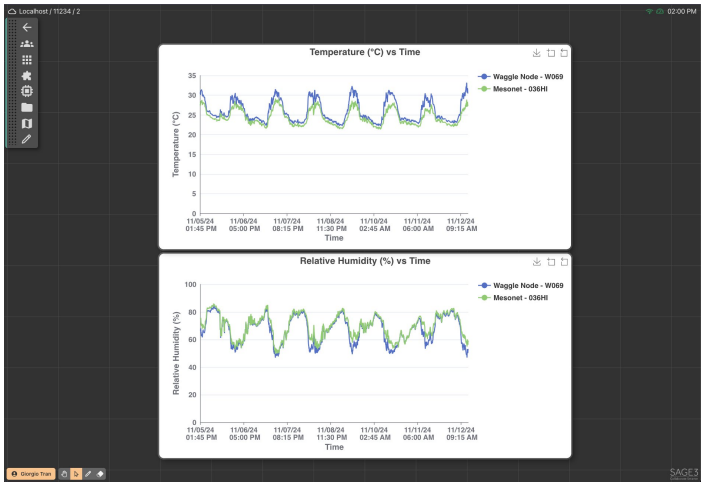

Figure 6: Generated visualizations displaying temperature and relative humidity data from the Pahoa Sage Node and Mesonet station in SAGE3.

3 Results

3.1 Fine Tuned YOLOv7

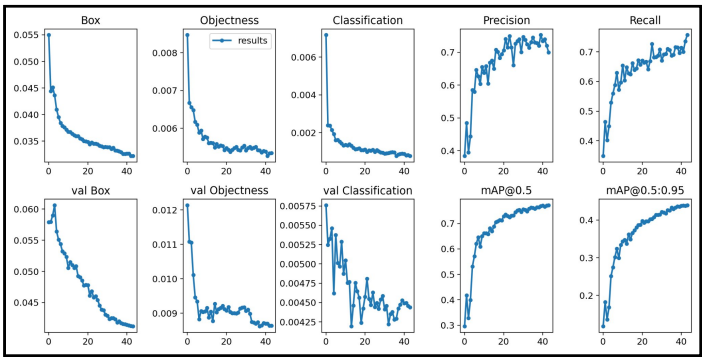

Figure 7: Training metrics showing loss reduction (box, objectness, classification) and performance improvement

(precision, recall, mAP@0.5, and mAP@0.5:0.95) over 43 epochs.

3.1.1 YOLOv7 Post-Fine Tuning Evaluation

Figure 7 shows the different components of loss in the YOLO model. The box loss represents the error in predicted bounding box coordinates for each detected object, while the objectness loss indicates whether an object is present within the bounding box. The classification loss measures the accuracy of object classification. The "val" versions of these losses represent evaluation metrics computed on the validation dataset, assessing how well the predictions align with the ground truth data.

In the training set, the losses for box, objectness, and classification converge, but in the validation set, the loss curves are more erratic, especially for classification. This could be due to class imbalance or a distribution shift between the training and validation sets.

Mean Average Precision (mAP) is a key metric for evaluating object detection models, measuring both object localization accuracy (through intersection-over-union with ground truth) and correct classification across multiple classes.

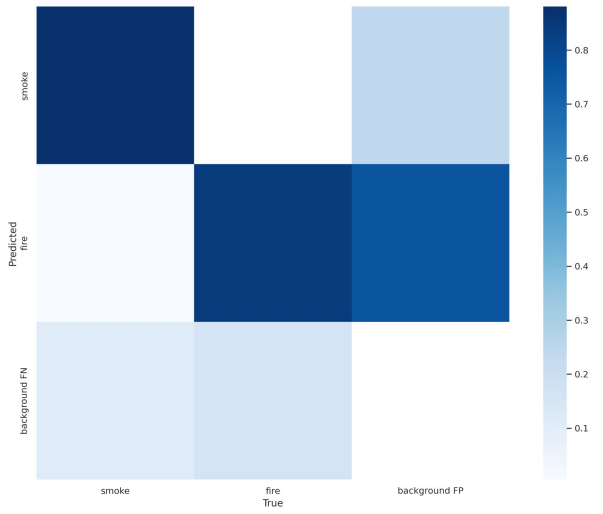

Figure 8: Confusion matrix on the validation set. 88% of predicted smoke instances were correctly classified as smoke, and 84% of predicted fire instances were correctly classified as fire. 76% of background instances were misclassified as fire, while 24% were misclassified as smoke. 16% of actual fire instances were not classified as fire, and 11% of actual smoke instances were not classified as smoke. Additionally, 1% of predicted fire instances were incorrectly classified as smoke.

mAP@0.5 evaluates model performance with a minimum 50% overlap between the predicted and ground truth bounding boxes, while mAP@0.95 requires 95% overlap. On the validation set, the model achieved an mAP@0.5 of 0.853 (Table 1), while on the test set, it scored 0.783 mAP@0.5 (Table 2) across all classes. The smoke class consistently outperformed the fire class in terms of mAP, which may be attributed to a class imbalance in the training data.

From the confusion matrices (Figures 8 and 9), it is evident that the model occasionally misclassified fire predictions. Specifically, there are instances where the model predicts fire when none is present, resulting in false positives. To address this issue, incorporating more examples of fire in the training dataset may help improve accuracy.

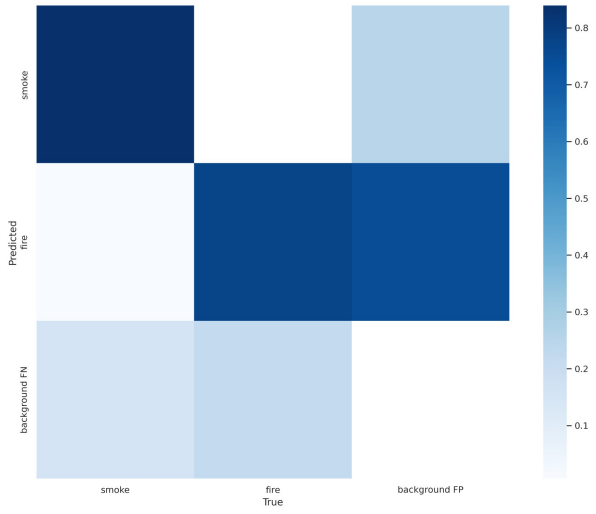

Figure 9: Confusion matrix on the test set. 84% of predicted smoke instances were correctly classified as smoke, and 78% of predicted fire instances were correctly classified as fire. 75% of background instances were misclassified as fire, while 25% were misclassified as smoke. 22% of actual fire instances were not classified as fire, and 15% of actual smoke instances were not classified as smoke. Additionally, 1% of predicted fire instances were incorrectly classified as smoke.

Overall, the model appears to be effective with a score of 0.783 mAP@0.5. Here is also a video of the model detecting fire from surveillance camera footage: https://drive.google.com/file/d/1AydCpPgMohvixcZ5F9PpcbBLj-qjawDJ/view?usp=drive_link

3.1.2 YOLOv7 Sage Node Deployment

The model successfully detected fire and smoke in an volcanic eruption image. It also identified smoke in instances where the camera was zoomed in on the eruption site. However, false positives occurred as well, such as a water droplet being misclassified as smoke (Figure 10B).

3.2 Starlink Traceroutes and Bandwidth

The traceroutes revealed a consistent pattern: regardless of the endpoint's location, packets would first pass through California before reaching their destination. This was also true for Hawaii, where the packet would travel to California before returning to the island.

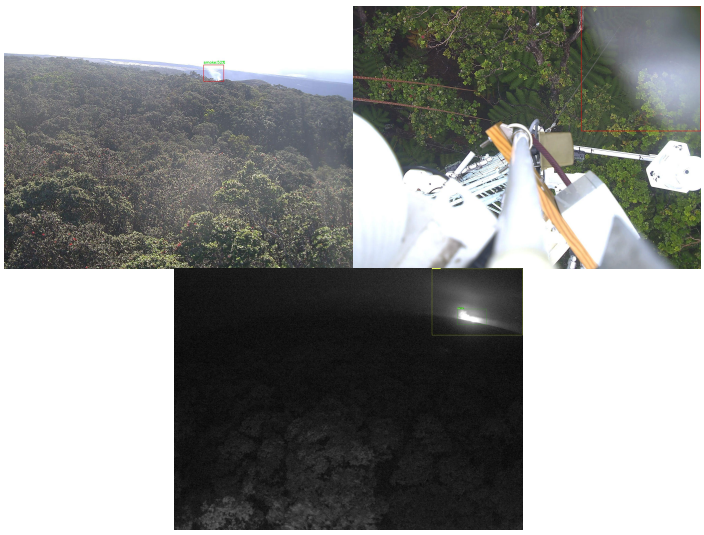

Figure 10: Smoke and fire detection from YOLOv7 model. (A) Smoke detected with the camera zooming towards Kilauea's location. (B) False positive from rain drop on camera lens. (C) Smoke and fire identified from the eruption image.

Bandwidth measurements from Ookla and the Starlink mobile app indicated download speeds of 200 Mbps and upload speeds of 30 Mbps. However, on the Lahaina Sage Node, the Starlink dish was misaligned by 26 degrees due to structural constraints of the mounting platform, resulting in download speeds of 30 Mbps and upload speeds of 5 Mbps.

The Starlink mobile application shows latency of 73 ms.

4 Discussion

4.1 Fine Tuned YOLOv7

The fine-tuned YOLOv7 model successfully detected fire and smoke in both the eruption image and an image showing smoke emitting from the volcano. However, it mistakenly classified a water droplet on the camera lens as smoke. This suggests that further training might be needed, and the dataset could benefit from additional images of artifacts that may impact the camera's lens.

4.2 Starlink Traceroutes and Bandwidth

The consistent routing pattern observed in the traceroutes—where packets always passed through California first, even for Hawaii to Hawaii communication—provides insight into how Starlink routes data. This suggests that the network infrastructure might prioritize certain regions or routing paths.

After the installation of the Lahaina Sage Node, suboptimal alignment of the Starlink dish led to a noticeable drop in performance. Unlike the tests conducted in the lab on Oahu, where the dish alignment was optimal, the misalignment at the Lahaina site resulted in significantly reduced download and upload speeds, with measurements of 30 Mbps and 5 Mbps, respectively. This highlights the importance of precise alignment for maintaining optimal Starlink performance.

Class	Precision	Recall	mAP@0.5	mAP@0.95
All	0.795	0.79	0.853	0.523
Smoke	0.845	0.834	0.892	0.577
Fire	0.746	0.747	0.815	0.469

Table 1: Precision, recall, mAP@0.5, and mAP@0.95 on validation set for the “Smoke” and “Fire” classes. The model performs better at detecting “Smoke” compared to “Fire” with higher precision, recall, mAP@0.5, and mAP@0.95. The “All” row represents the combined performance for both classes.

Class	Precision	Recall	mAP@0.5	mAP@0.95
All	0.74	0.734	0.783	0.434
Smoke	0.791	0.782	0.832	0.492
Fire	0.688	0.685	0.733	0.377

Table 2: Precision, recall, mAP@0.5, and mAP@0.95 on the test set. The model performs better at detecting “Smoke” compared to “Fire” with higher precision, recall, mAP@0.5, and mAP@0.95. The “All” row represents the combined performance for both classes.

4.3 Ramifications for Emergency Response Data in Hawaii

Routing traffic through California introduces unnecessary latency, as data must travel to the continental US and back. This delay can be critical in emergency response scenarios requiring real-time or near-real-time data transmission. Furthermore, it can strain bandwidth during peak usage, potentially slowing data transmission for emergency detection and response systems. During hurricane Dora, winds in Lahaina reached speeds of approximately 30 m/s downslope [2]. Using the 10% wind speed rule of thumb for fire spread estimation [6], the fire’s rate of spread can be approximated at 10.8 m/s. With a 73 ms latency via Starlink, the fire would spread about 0.79 meters during this delay. Because the upload speed at the Lahaina Sage Node via Starlink is 5 Mbps, uploading a 2 MB image would take at least 3 seconds. In that time, the fire could spread approximately 32.4 meters, factoring in the estimated rate of fire spread. This calculation does not account for additional delays from processing models hosted in the cloud instead of at the edge. Such latency emphasizes the importance of edge computing, where analysis is performed locally, reducing delays and enabling faster response in critical situations.

Despite these challenges, Starlink remains a valuable asset in local emergencies, as it lessens reliance on cellular infrastructure, which in the case of Lahaina, was burned down during the fire.

5 Conclusion

Edge sensors with GPUs have the potential to transform wildfire detection and monitoring by enabling rapid early analysis and detection without relying on cloud processing. When combined with Starlink, these edge sensors become especially valuable for disaster detection in remote areas where traditional connectivity is limited, or in the case of Lahaina where cellular communication towers were burned down. The data visualizer in SAGE3 provides users with a unifying interface to monitor and compare data across Sage nodes and Mesonet sensors.

6 References

[1] P. Beckman, R. Sankaran, C. Catlett, N. Ferrier, R. Jacob and M. Papka, "Waggle: An open sensor platform for edge computing," 2016 IEEE SENSORS, Orlando, FL, USA, 2016, pp. 1-3, doi: 10.1109/ICSENS.2016.7808975.

[2] Cruz, M.G., Alexander, M.E. The 10% wind speed rule of thumb for estimating a wildfire's forward rate of spread in forests and shrublands. Annals of Forest Science 76, 44 (2019). https://doi.org/10.1007/s13595-019-0829-8

[3] IP2Location. ip2location.io

[4] IPInfo. ipinfo.io

[5] IPGeolocation. ipgeolocation.io

[6] Juliano, T. W., Szasdi-Bardales, F., Lareau, N. P., Shamsaei, K., Kosović, B., Elhami-Khorasani, N., James, E. P., and Ebrahimian, H.: Brief communication: The Lahaina Fire disaster – how models can be used to understand and predict wildfires, Nat. Hazards Earth Syst. Sci., 24, 47–52, https://doi.org/10.5194/nhess-24-47-2024, 2024.

[7] SAGE3. sage3.app

[8] de Venâncio, P.V.A.B., Lisboa, A.C. & Barbosa, A.V. An automatic fire detection system based on deep convolutional neural networks for low-power, resource-constrained devices. Neural Comput & Applic 34, 15349–15368 (2022). https://doi.org/10.1007/s00521-022-07467-z

[9] C. -Y. Wang, A. Bochkovskiy and H. -Y. M. Liao, "YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors," 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 2023, pp. 7464-7475, doi: 10.1109/CVPR52729.2023.00721.

7 Acknowledgments

This project is funded in part by National Science Foundation awards: 2346568, 2149133, 2004014, 2003800, 2003387.