Applied Computer Science 2026, vol. 22, no. 1, pp. 110–122
https://doi.org/10.35784/acs_8125

Submitted: 2025-07-20 | Revised: 2025-12-05 | Accepted: 2025-12-12

CC-BY 4.0

Keywords: forest fire detection, convolutional neural networks (CNN), transfer learning, parallel processing, edge computing

Achraf Nasser Eddine BELFERD [1*], *Hamdan BENSENANE* [1], *Abdellatif RAHMOUN* [1]

[1] LabRI-SBA Lab., Ecole Superieure en Informatique, Sidi Bel Abbes, Algeria, a.belferd@esi-sba.dz, h.bensenane@esi-sba.dz, a.rahmoun@esi-sba.dz

[*] Corresponding author: a.belferd@esi-sba.dz

A scalable and cost-effective forest fire detection approach using deep transfer learning on a Raspberry Pi cluster

Abstract

Due to the increasing frequency of forest fires and their rapid spread, early detection is critical for effective containment and mitigation. This paper proposes a cost-effective edge-based forest fire detection system that receives images from multiple sources (terrestrial cameras and UAVs) to predict and alert authorities of potential forest fires. The model of the system is built using transfer learning with MobileNetv2 on a realistic and diverse dataset, resulting in a lightweight CNN model that is further optimized by using quantization to reduce its size and improve the inference speed. The proposed model is deployed on an 8-node Raspberry Pi cluster, using Slurm and MPI to manage cluster task scheduling and parallel processing. The proposed system achieves 99.21% accuracy, precision, recall, and F1 score on a realistic test dataset containing vague real-world scenarios such as fog and sunset conditions, with an inference speed of 69 frames per second. These results, along with the system's autonomous and offline operation, cost effectiveness, power efficiency, and scalability, make it ideal for real-time forest fire monitoring at the edge, even in off-grid and remote areas.

1. INTRODUCTION

Forest fires are among the most severe natural disasters, caused by human activities, natural causes (lightning, volcanic eruptions) and, more recently, climate change, which has made landscapes more fire-prone, with an increase in the area burned, especially in high-latitude forests (Jones et al., 2022; 2024; Zheng et al., 2023). These fires pose significant threats to ecosystems and biodiversity, while also contributing to global warming and climate change through the release of significant amounts of greenhouse gases and particulate matter (Singh, 2022).

As a countermeasure, an increasing number of studies have focused on the early detection of forest fires (Barmpoutis et al., 2020), some of which use wireless multimodal sensors (heat, smoke, humidity, etc.). However, despite their advantages, these approaches face many challenges related to power consumption, computational requirements, and the delay caused by the time required for particles to reach the small detection range and activate the sensors, requiring a large number of deployments to cover large areas (Aslan et al., 2012; Bouabdellah et al., 2013; Dampage et al., 2022).

In contrast, recent research has taken advantage of deep learning's ability to extract and learn complex feature representations to achieve high image classification performance by training a convolutional neural network (CNN) with different datasets and then using the model to analyze surveillance footage from fixed satellites, cameras, and UAVs (Q. Zhang et al., 2016; Seydi et al., 2022; Shamta & Demir, 2024). Building on this, transfer learning, which takes the weights of pre-existing models (usually feature extraction layers) and then modifies the output layer before retraining the newly constructed model on a specified custom dataset, has allowed researchers to overcome the lack of available datasets and improve detection accuracy (A. Khan et al., 2022; Yandouzi et al., 2022; Yang et al., 2023).

However, despite the promising results of these studies, many rely on resource-intensive hardware or cloud-based solutions that limit their use in remote, off-grid forests. In addition, there is sometimes a significant delay between fire outbreak and detection, especially when using satellite imagery or cloud services, making

the system more suitable for damage assessment than real-time detection. In addition, variations in environmental conditions such as sunset and fog could be mistaken for fire or smoke, highlighting the need for a robust, adaptable solution.

To address some of these limitations, the use of low-cost edge computing devices such as Raspberry Pi offers a promising alternative. The Raspberry Pi model 4, with a quad-core ARM Cortex-A72 CPU running at 1.8 GHz and up to 8 GB of DDR4 RAM (Raspberry Pi Foundation, n.d.), could provide sufficient computing power to run optimized CNN models. In addition, its compact size, low power consumption, and low cost make it particularly suitable for deployment in remote areas. Moreover, the formation of a cluster of Raspberry Pi nodes would result in easier management and scalability of processing by enabling parallel inference on incoming data (Marković et al., 2019). This enables distributed processing of images in real time, improving responsiveness and extending coverage to multiple monitoring sites simultaneously, while maintaining offline operational autonomy even in off-grid areas.

In this paper, we propose a cost-effective and scalable forest fire detection system using a lightweight CNN model deployed on an 8-node Raspberry Pi cluster, where Slurm and MPI are used for task scheduling and parallel processing. The proposed model is built using transfer learning with MobileNetV2, trained on a rich dataset accounting for changing environmental conditions such as sunset and fog, and then optimized for edge inference through quantization. This system enables real-time image-based fire detection from multiple sources with high accuracy while maintaining autonomy, energy efficiency, and low deployment cost. Thus, a key contribution of this work is the integration of a lightweight transfer learning CNN with distributed inference on a Raspberry Pi cluster, an architecture not previously explored for forest fire detection. The rest of this paper is organized as follows: Section 2 reviews related work, Section 3 describes the methodology, Section 4 presents experiments and results, Section 5 discusses the results, and finally Section 6 concludes.

2. RELATED WORKS

In the last decade, extensive studies have been conducted on forest fire detection, where different approaches have been explored and developed to provide effective solutions. For the sensor-based approach, Aslan et al. (2012) proposed a comprehensive framework including a wireless sensor network (WSN) architecture, deployment scheme, and clustering and communication protocols, which resulted in effective fire detection with minimized energy consumption, but they faced challenges related to the high cost of battery replacement and recharging and the difficulty of deployment. Similarly, AA Alkhatib (2013) used WSN for fire detection in which temperature sensors operate in swarms, as the detection of high temperature by a node alerts and notifies the surrounding nodes to collect more information. Expanding on this, Dasari et al. (2020) proposed an automated detection system based on both fire and smoke sensors, where the master node uses radio frequency for communication, analyzes the signals sent by the slave nodes, and then uses GSM to alert the base. Moreover, Dampage et al. (2022) introduced the use of a machine learning regression model to improve the accuracy of their WSN fire detection system, reaching a theoretical accuracy of 81%, but the use of this method requires considerable computational power in addition to the already existing limitations of using sensors. Furthermore, Apriani et al. (2022) proposed a fire detection system based on LoRa as a wireless network and an Arduino Uno controller to regulate the inputs of thermal AMG8833 sensors, and despite their accuracy and high communication range, which reached up to 500 meters, the low detection range of the sensors requires the deployment of numerous sensors to cover large areas.

While sensor-based systems have provided valuable advantages in early detection and modular deployment, they typically suffer from critical limitations such as the sensor's low detection range, high deployment density requirements, and power consumption and supply.

To address some of the limitations of the sensor-based system, Q. Zhang et al. (2016) used deep learning by implementing a CNN classifier model to detect fire patches, resulting in 90% accuracy on the test dataset. Building on this, Seydi et al. (2022) proposed Fire-Net, a deep learning framework trained on satellite imagery to detect forest fires. Despite the accuracy of their system at 97.35%, the reliance on satellite imagery as an input limits the responsiveness and availability of the system. To address the delay limitation, Kang et al. (2022) used the Himawari-8 Advanced Himawari Imager, which has a 10-minute interval, along with a CNN model classification to detect wildfires with high accuracy, but with a limited coverage problem and the risk of false alarms. Furthermore, Shamta and Demir (2024) proposed an aerial CNN-based fire detection system,

where a UAV equipped with a camera and NVIDIA Jetson Nano as an embedded AI computer is tasked to monitor a specific area and perform real-time inference.

Despite the promising performance of these deep learning-based systems, they often require large and high-quality labeled datasets to perform well under various real-world scenarios.

To overcome this limitation, the use of transfer learning was an appealing solution, as A. Khan et al. (2022) developed the Deep Fire system, which uses a VGG19-based transfer learning approach executed in the cloud to perform inference on the images captured by UAVs. Similarly, but using other CNN models as the base model for transfer learning, a high accuracy of 99% was demonstrated by Yandouzi et al. (2022) in their work with the implementation of a UAV for monitoring and a cloud service for performing the inference. In addition, L. Zhang et al. (2022) also focus on the use of UAVs as monitors and propose the FT-ResNet50 model using transfer learning on the ResNet pre-trained model. Building on this, Yang et al. (2023) introduced a UAV-centric forest fire smoke detection system using transfer learning on their refined YOLOv5 CNN, resulting in an 11.1 million parameter model and an accuracy of 96%, but as a common drawback like the previous works, the system would mistake the natural environment as a fire outbreak in some cases. Another notable contribution is the work of Reis and Turk (2023), as they achieved an accuracy of 99.32% by using transfer learning on the DenseNet121 model with the FLAME dataset (Shamsoshoara et al., 2020). Furthermore, more studies are still adding to this field, as recently Yunusov et al. (2024) introduced a fire detection system using transfer learning on YOLOv8 alongside TranSDet with a training dataset of 5200 images, achieving a result of 97%, but they discovered that the system can't detect smoke in addition to confusing sunlight with fire. Similarly, (Gupta & Mishra, 2024) had impressive results for forest fire detection using transfer learning with pre-trained MobileNetV2, where they achieved an accuracy of almost 99%, but this work didn't discuss the deployment of their system, leaving a research gap to consider cost, energy constraints, network independence, or the type of implementation architecture (centerized/distributed) required for remote forest areas.

Despite the high performance of these deep learning and transfer learning based systems, several limitations remain as many of them rely on cloud infrastructure or high performance devices, which may not be deployable in remote off-grid forest environments due to energy consumption issues and lack of reliable internet connectivity. In addition, their reliability is affected by misclassification due to environmental changes (sunlight, fog). These limitations represent key research gaps that our work aims to address, as detailed in the following sections.

To overcome the energy consumption, computational requirements of standalone devices, and connectivity limitations of cloud-based systems in other domains, some researchers have explored the use of edge single board computer (SBC) clusters. As Baun, (2016) demonstrated by building a cluster of 8 Raspberry Pi model B and highlighting the cost-effectiveness along with energy efficiency that these clusters can offer, making them suitable for academic projects and research. Expanding on this topic, (Johnston et al., 2018) presented various benefits of using SBC clusters such as energy efficiency, cost, redundancy, scalability, parallelization, and management, concluding that they are suitable for many modern use cases such as edge computing. Based on this, Fernández-Cerero et al. (2019) proposed a cost-effective model based on a low-cost SBC cluster, with the aim of using it as an alternative to fog computing to achieve a scalable system that's suitable for the requirements of real-time IoT tasks. Similarly, M. N. A. Khan et al. (2019) developed a low-cost flame detection system using Raspberry Pi and classical image processing methods (RGB, HSV, LAB filtering).While it has the advantage of being low-cost, the detection approach that relies on color-based heuristics suffers from high false positive rates and poor robustness to environmental variations, making it not applicable for outdoor use, such as in the case of forest fire detection, where lighting changes, shadows, and reflections are common. Moreover, in the same area, Srinivas et al. (2024), also proposed an indoor alert system based on Raspberry Pi, and trained both YOLOv4-tiny and YOLOv5s models for fire and smoke detection, which achieved reasonable mAP values (42.3% for fire and 35. 4% for smoke), however, it was limited to indoor environments and only operated on a single Raspberry Pi, making their system unsuitable for monitoring large environments, and also doesn't address scalability for larger workloads. Furthermore, in our previous work Belferd and Rahmoun (2022), we analyzed the performance and different uses of SBC clusters, confirming their viability for distributed parallel processing at the edge (Cox et al., 2014; Diwedi & Sharma, 2018; Kanani & Padole, 2020).

Motivated by these limitations, this work proposes a novel forest fire detection system by integrating transfer learning with a lightweight CNN and deploying it on a multi-node Raspberry Pi cluster for scalable edge inference. Unlike existing approaches that rely on single-board processing or cloud offloading, the proposed approach performs real-time distributed processing using Slurm and MPI, enabling faster inference,

larger spatial coverage, and fully offline operation without sacrificing high forest fire detection accuracy. To the best of our knowledge, no previous study has combined transfer learning with a cluster-based SBC architecture for forest fire detection, as this work addresses a critical gap in the literature by demonstrating a practical, cost-effective, scalable, and autonomous system suitable for remote forest environments.

3. METHODOLOGY

This research aims to develop a forest fire detection system at the edge using Deep Transfer Learning on a pre-trained CNN model to create our own lightweight and efficient forest fire detection model, which would be deployed on an 8-node Raspberry Pi 4 cluster at the edge. The proposed system receives images from multiple sources (terrestrial cameras, Unmanned Aerial Vehicles), stores them in a shared network file system, then using the computational power of the cluster, it runs these images through our classification model to detect if any of them has a potential forest fire, where in this case the image along with its source ID/coordinates are sent to the fire department for validation and take appropriate action Figure 1.

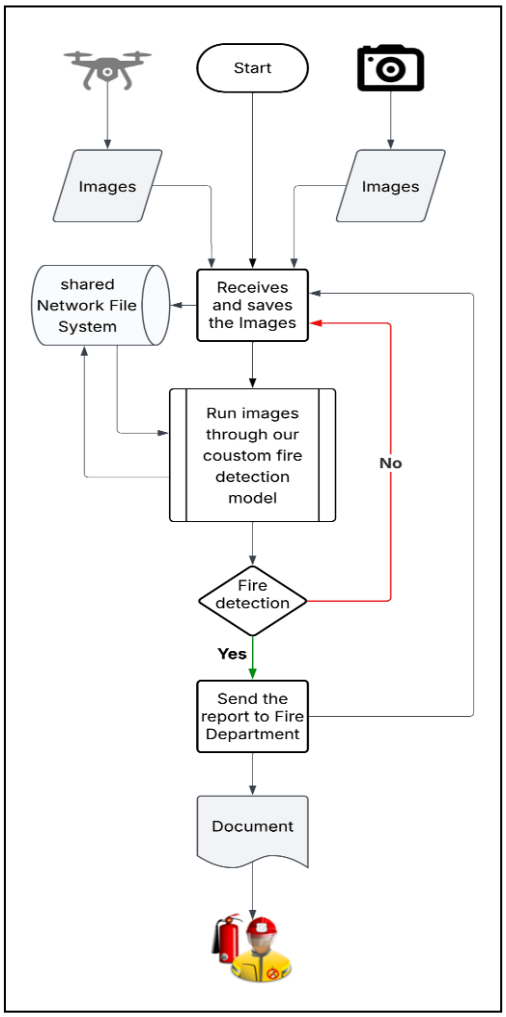

Fig. 1. The proposed system overview workflow diagram

3.1. System hardware

Our System, as shown in Figure 2, includes:

- A cluster of 8 Raspberry Pi 4 nodes is used, where we dedicated a master/controller node to manage the other 7 worker nodes.
- A USB drive is mounted on the master node as a shared Network File System.
- A router to ensure communication.

– RJ45 cables.

Fig. 2. System hardware

3.2. Dataset preparation

We used the Forest Fire Dataset (Alik05, 2022) for our model training and testing. It is a balanced binary dataset of 1900 real images with no synthetic, rendered, or simulated images. photographs. This dataset is divided into 950 images belonging to the fire class and the other 950 images belonging to the no fire class. After several tests, we found the best results after using a combination of stratified splitting to maintain class balance and a group-based splitting strategy to prevent data leakage from image sequences or scene overlap, along with the use of 64% (1216 images) for training, 16% (304 images) for validation, and the last 20% (380 images that weren't included in training or validation) for testing. In addition, data augmentation and transformation techniques such as random flipping, rotation, and brightness adjustment are used to further enrich the dataset.

The reasons for selecting this dataset are: the size of the dataset (which was large enough after using data transformation and augmentation for training our model to obtain effective results) and the variety of colorful real images that well represent a forest fire, Figure 3. And finally, it contains images with real-life scenarios such as high-density fog and sunset that could be mistaken for smoke or fire, Figure 4.

Fig. 3. Forest fire images from the dataset

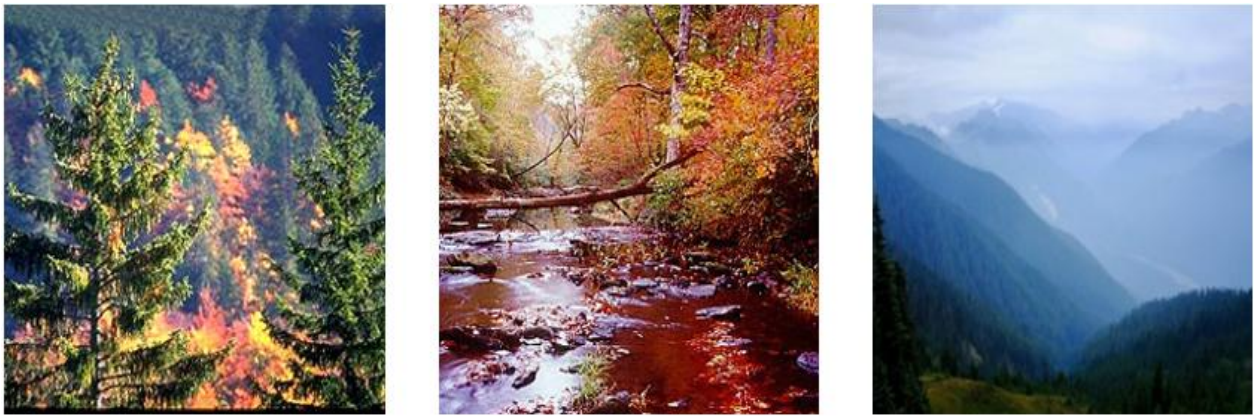

Fig. 4. Forest images that could be mistaken for a forest fire

3.3. Model architecture

There are many CNN models, each with different characteristics and effectiveness Table 1. In our case, since we are working with Raspberry Pi 4 boards, some constraints are imposed, which is the reason why some models are more suitable for our cluster than others. That's why we chose Mobilenetv2 as our base model, taking advantage of: low number of parameters, low latency and small size.

Using Deep Transfer Learning with the pre-trained model Mobilenetv2 by removing the original classification layers and using the dropout technique by freezing the feature extractor. and then adding pooling and a dense layer to classify the output into two categories: Fire and No Fire Figure 5. This results in a total of 2,260,546 parameters, of which 2,257,984 are not trainable (because we froze them) and 2,562 are trainable using the dataset.

Tab. 1. Different CNN models specifications

Model	Size (MB)	Params (Millions)	Latency (CPU)
MobileNet	~16 MB	~4.2M	Very Fast
MobileNetV2	~14 MB	~3.4M	Very Fast
NASNetMobile	~23 MB	~5.3M	Moderate
EfficientNetB0	~29 MB	~5.3M	Moderate
DenseNet121	~33 MB	~8M	Slower

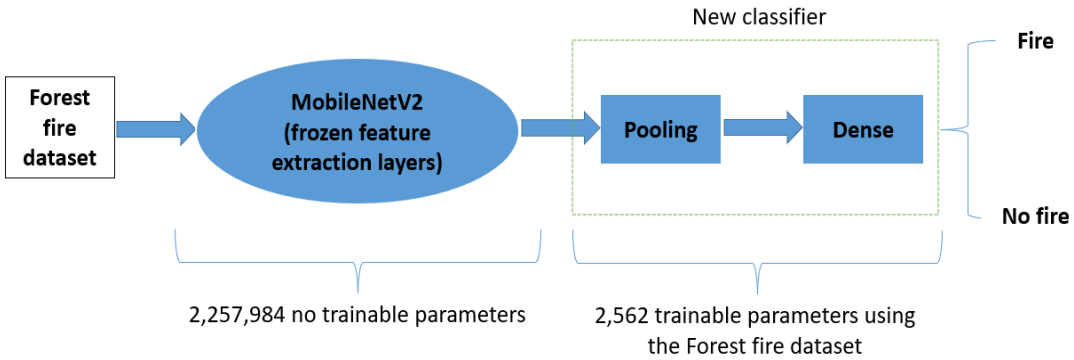

Fig. 5. Our system classification model building using deep transfer learning

3.4. Model deployment

After we finish training the model, we export it as a TFLite file, which is suitable for our use case at the edge, since it offers the following features:

- Speed optimization (using int8 quantization to trade <2% accuracy for faster inference).
- Smaller model size.
- Lower power consumption.
- Requires only the TFLite interpreter, which is lightweight compared to running full TensorFlow.

This model is then deployed on the USB drive used as a shared network file system, along with the Python environment and script used by our cluster system.

3.5. System execution

First, the master node receives a new set of images from several predefined sources, stores the images in the shared network file system, and then uses Slurm as a scheduler that calls mpi4py (Message Passing Interface, which ensures communication between cluster nodes during parallel execution) to assign ranks to each available process (in this case, to each of the 4 CPU cores of the available worker nodes). Then, each process loads its own copy of the model so that it can use its own isolated interpreter instance. At the same time, the rank 0 process divides the images into equal chunks and sends each process its assigned image paths, allowing each process to start the inference task in parallel with the other processes. After each process completes its inference task, the list of predicted classes and ground truth is built and sent back to the rank 0 process, where the results are aggregated and a report is generated. Finally, the master node checks the report for any predicted fire image, so it sends a notification to the fire department containing the image and its source location Figure 6.

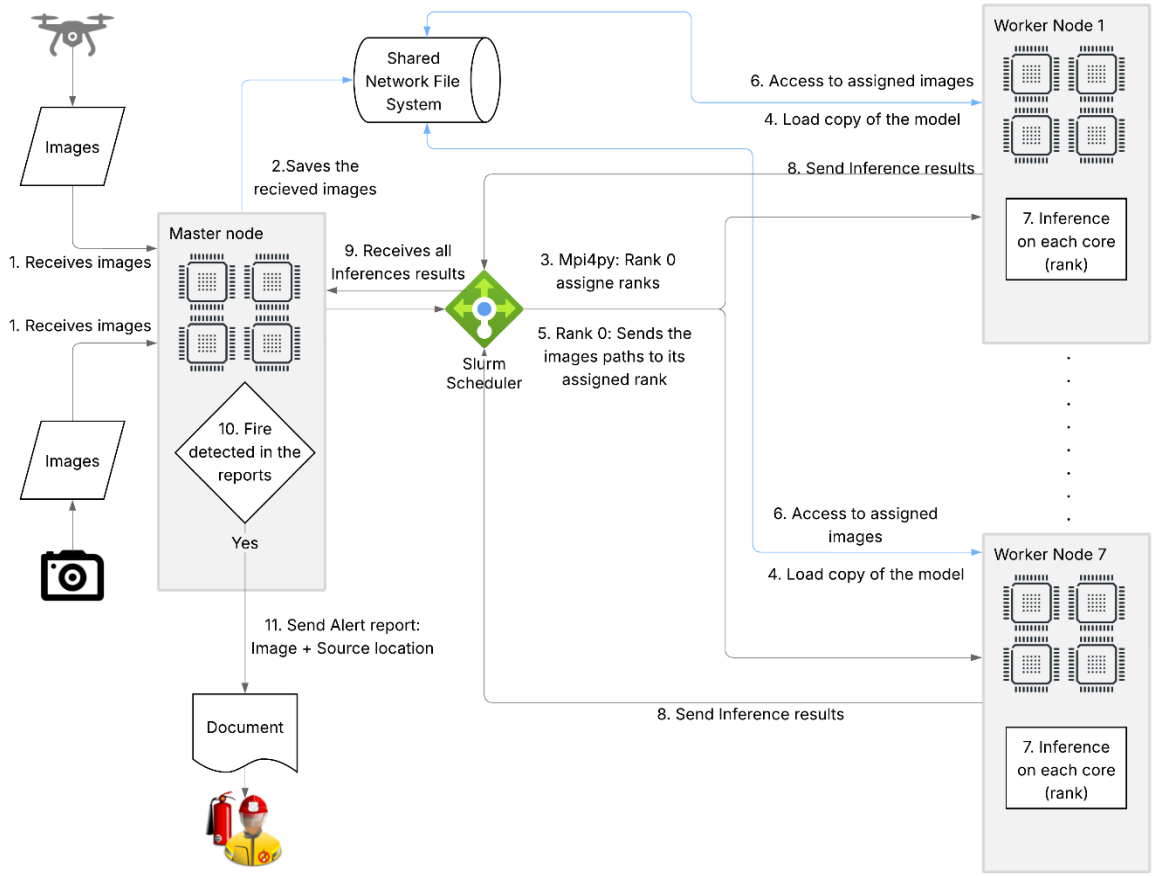

Fig. 6. System execution

4. EXPERIMENTS AND RESULTS

Before evaluating our fire detection system, and to ensure the cluster's scalability and efficient use of its resources (using many nodes in the cluster isn't a waste), we perform classic parallelism tests to evaluate the cluster's performance (execution time) versus the difficulty of the task while varying the number of worker nodes.

The first test consists of finding the prime numbers between 2 and N in parallel (where we increase N each time to see how the cluster responds to a larger workload). We observe that the execution time has a negligible difference for lower values of N (1000, 10000, 100000), even when using more cluster nodes, as shown in Figure 7. However, after reaching a higher value of N (more than 100000), the use of more nodes resulted in lower execution times, which scaled positively, especially for larger N values.

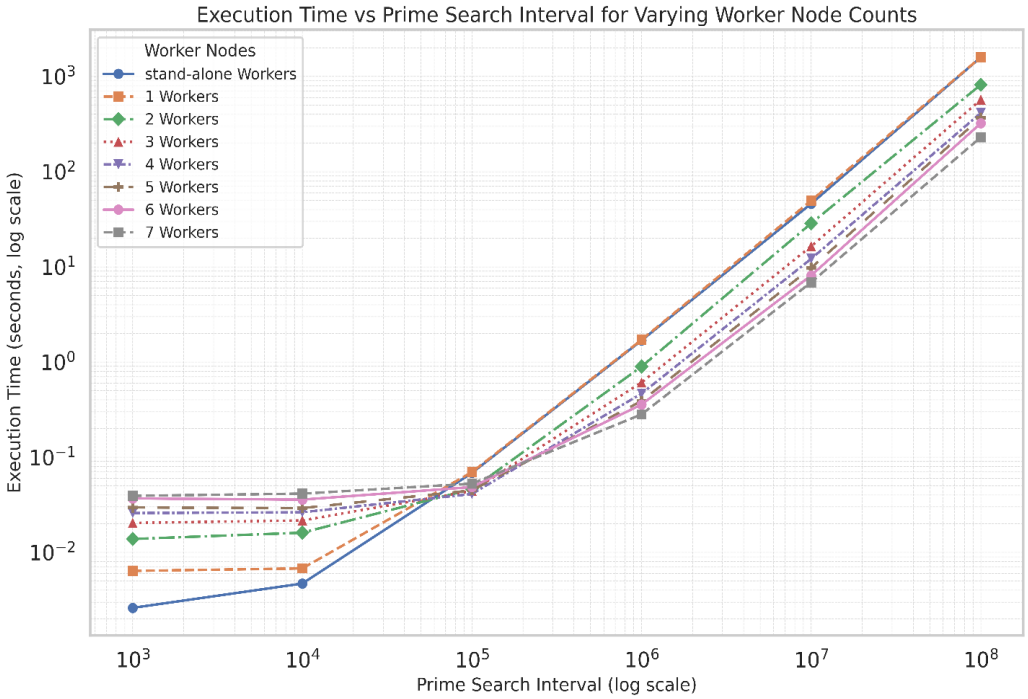

Fig. 7. Execution time vs prime search interval for varying worker node counts

The second test performs a parallel approximation of the π value, which is a demo program provided in the mpi4py repository (US Army Engineer Research and Development Center, n.d.) to test the parallel execution of the cluster, using the midpoint rectangle method with *n* intervals (1). We chose this test because of its slow convergence at higher *n* values, which requires more computing power to stress test our cluster, as observed in the test results Figure 8, where the use of more nodes resulted in lower execution time only after reaching N = 100000, while before that interval the use of one node performed better.

$$\pi \approx h * \sum_{i=1}^{n} \left[\frac{4}{(1 + x_i^2)} \right] \tag{1}$$

$$h = \frac{1}{n} \tag{2}$$

$$x_i = h * (i - 0.5) \tag{3}$$

Where: n – number of intervals (the higher, the more accurate and the more calculations required).
h – width of each interval.
x_i – midpoint of the i-th interval.

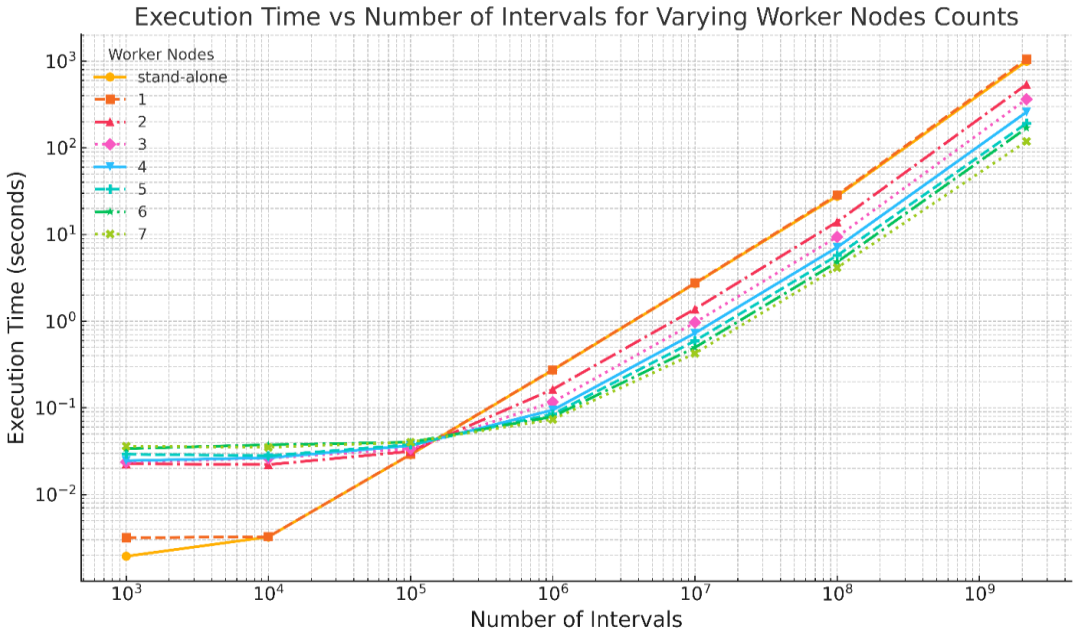

Fig. 8. Execution time vs number of intervals for varying worker node counts

As for our forest fire detection CNN model training, showed increasing values with fast convergence over the 8 epochs (where early stopping was used to prevent overfitting) in both training and validation accuracy until they reached 0.99 and 0.98, respectively, Figure 9. Also, the model training and validation loss curves over the epochs continued to converge Figure 10, which, together with the earlier accuracy results, indicate no signs of overfitting.

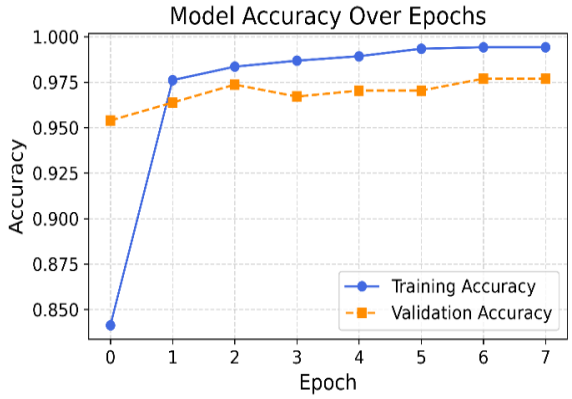

Fig. 9. Model training and validation accuracy

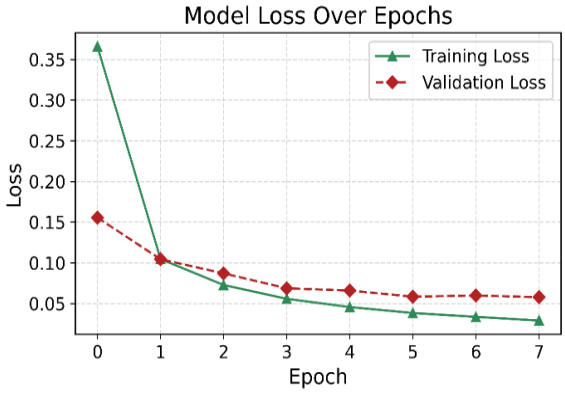

Fig. 10. Model training and validation loss

Finally, on the test dataset, the proposed model achieved high classification performance results as shown in the confusion matrix Figure 11 and more details in Table 2, where an overall accuracy of 0.99 was achieved with high precision, recall and F1 score for both classes.

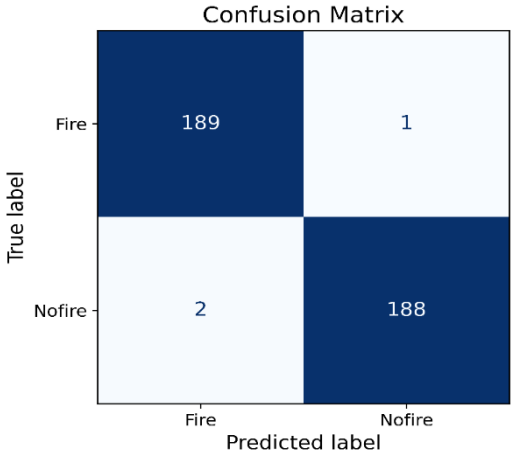

Fig. 11. Test confusion matrix

Tab. 2. Model test classification results

Class	Precision	Recall	F1-Score	Support
Fire	0.9895	0.9947	0.9921	190
Nofire	0.9947	0.9895	0.9921	190
Accuracy	/	/	0.9921	380
Macro Avg	0.9921	0.9921	0.9921	380

In addition, Figure 12 demonstrates the scalability of the system, showing that the execution time decreases as the number of worker nodes increases. In addition, when using all 8 nodes of the cluster, the execution time was close to that of the PC with Intel 12400f CPU, 16GB RAM, and 1TB NVMe M.2 SSD.

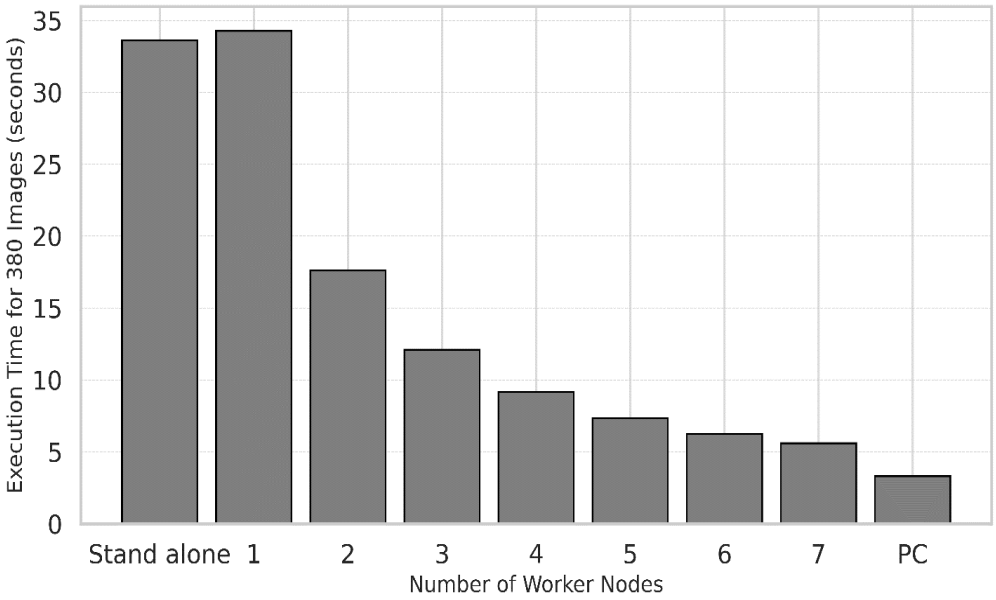

Fig. 12. Execution time vs number of worker nodes

5. DISCUSSION

The cluster performance scaling tests for both prime number search and π value approximation showed that for small tasks (low values of *N*), the execution time using a single Raspberry Pi was slightly better than using multiple nodes of the cluster, and after performing various tests, we found that the time spent in the communication and MPI overhead processes was greater than the actual computation process, hence the higher execution time when using multiple nodes. As a result, to maximize the use of cluster resources, the task must

be large enough according to the number of nodes used in the cluster, otherwise the communication time may outweigh any gains from using parallelization, as shown in equation (4).

$$Parallelism\_Worth\ (S,N) = \frac{T_{serial}(S)}{T_{comp}(S,N) + T_{comm}(N)} \tag{4}$$

where: N – number of worker nodes.
S – size of the task (as for our examples, number of intervals, number of images).
$T_{serial}(S)$ – execution time of the task on a single node.
$T_{comp}(S,N)$ – computation time of the task on N nodes.
$T_{comm}(N)$ – communication time when using N nodes.

In terms of model performance, the proposed CNN model achieved high results in all metrics: size, accuracy, recall, precision, and F1-score compared to other general CNN models that we tested Table 3, and to the results of (A. Khan et al. 2022), where our model performed better on the same dataset they used. Furthermore, deploying the model on the 8 node cluster proved to benefit from the scalability of the system by having better execution times with more worker nodes, as we noticed that 380 test images were processed in 5.6 seconds when all nodes are used, which results in approximately 67 images/second, this means that the system can cover almost 67 sites simultaneously in real time, or we can trade the real time processing to cover more sites, for example for 10 seconds delay the system can cover up to 670 sites without risking a large wildfire outbreak.

Tab. 3. CNN Models test results

Model	Accuracy	Precision	Recall	F1-Score	Params (Millions)
MobileNet	0.500	0.250	0.500	0.333	~4.2M
MobileNetV2	0.500	0.250	0.500	0.333	~3.4M
NASNetMobile	0.718	0.721	0.718	0.712	~5.3M
EfficientNetB0	0.500	0.250	0.500	0.333	~5.3M
DenseNet121	0.805	0.844	0.805	0.799	~8M
(A. Khan et al., 2022)	0.950	0.957	0.942	0.949	(not mentioned)
Proposed model	0.990	0.990	0.990	0.990	~2.26

Moreover, taking into account the cost of the system, where each Raspberry Pi 4 (4GB) model costs about 55$ (Newark Electronics, n. d.) and a PoE 8-port switch for $60 (or a regular 8-port switch for $20), this would result in a total cost of $420-460, and a value of 4.25 performance per dollar equation (5), in addition to the power consumption of 6.4W per node when the system is underloaded, makes the proposed system outperform the PC used in our experiments in terms of cost effectiveness, power efficiency, deployment flexibility, scalability, and management, making this system ideal for the edge environment.

$$Performance\ Per\ Dollar = \frac{1}{Execution\ Time \times System\ Cost} \tag{5}$$

Finally, the contribution of this work is the integration of a lightweight transfer learning CNN with distributed inference on a Raspberry Pi cluster. Which is an architecture not previously implemented for forest fire detection has demonstrated its cost-effectiveness and scalability to achieve real-time forest fire detection or to trade off system responsiveness for greater coverage. However, despite its promising gains, the system has certain limitations regarding the installation process, the type of network used, and the power source, where a renewable energy source (solar, wind, etc.) could be used to further enhance the system's autonomy and existence in off-grid forest regions that may lack reliable conventional power infrastructures.

6. CONCLUSION

This work presents a cost-effective, scalable forest fire detection system at the edge using transfer learning on the MobileNetV2 CNN. After training on a realistic and diverse dataset for this use case, we achieved a lightweight yet powerful classifier with 99.21% accuracy, recall, precision, and F1 score. In addition, tests

were performed on an 8-node Raspberry Pi 4 cluster, proving the benefits of cost-effectiveness, low power consumption, and scalability, making it ideal for our model deployment at the edge, where the proposed system demonstrated the ability to cover over 67 locations simultaneously with a real-time performance of approximately 67 images/second. In addition, a proposal is made to reduce the cost of expanding coverage by reducing the responsiveness of the system to cover more sites, as for a 10-second delay, the system could cover 670 sites without risking an impactful outbreak of wildfire. However, despite the promising gains, there are some challenges related to the initial installation, the type of network used, and the power source. To address this, we propose the use of renewable energy to further extend the reach of the system to off-grid forest regions where reliable conventional energy infrastructure is lacking. As part of future work, we aim to integrate complementary sensor modalities (temperature, gas, humidity) to increase the robustness of detection and reduce the reliance on vision-based inputs alone for decision making.

Funding

This research was funded by LabRI-SBA Laboratory at Ecole Supérieure en Informatique, Sidi Bel Abbes, Algeria

Conflicts of Interest

The authors confirm that they have no affiliations with or involvement in any organization or entity that has a financial or non-financial interest in the subject matter or materials discussed in this manuscript.

REFERENCES

AA Alkhatib, A. (2013). Smart and low cost technique for forest fire detection using wireless sensor network. *International Journal of Computer Applications*, *81*(11), 12–18. https://doi.org/10.5120/14055-2044

Alik05. (2022, April 11). *Forest Fire Dataset*. Retrieved July 15, 2025 from https://www.kaggle.com/datasets/alik05/forest-fire-dataset

Apriani, Y., Oktaviani, W. A., & Sofian, I. M. (2022). Design and implementation of lora-based forest fire monitoring system. *Journal of Robotics and Control (JRC)*, *3*(3), 236–243. https://doi.org/10.18196/jrc.v3i3.14128

Aslan, Y. E., Korpeoglu, I., & Ulusoy, Ö. (2012). A framework for use of wireless sensor networks in forest fire detection and monitoring. *Computers, Environment and Urban Systems*, *36*(6), 614–625. https://doi.org/10.1016/j.compenvurbsys.2012.03.002

Barmpoutis, P., Papaioannou, P., Dimitropoulos, K., & Grammalidis, N. (2020). A review on early forest fire detection systems using optical remote sensing. *Sensors*, *20*(22), 6442. https://doi.org/10.3390/s20226442

Baun, C. (2016). Mobile clusters of single board computers: An option for providing resources to student projects and researchers. *SpringerPlus*, *5*(1). https://doi.org/10.1186/s40064-016-1981-3

Belferd, A. N. E., & Rahmoun, A. (2022). A single board computer and its cluster applications. *2022 2nd International Conference on Advanced Electrical Engineering (ICAEE)* (pp. 1–3). IEEE. https://doi.org/10.1109/icaee53772.2022.9962136

Bouabdellah, K., Noureddine, H., & Larbi, S. (2013). Using wireless sensor networks for reliable forest fires detection. *Procedia Computer Science*, *19*, 794–801. https://doi.org/10.1016/j.procs.2013.06.104

Cox, S. J., Cox, J. T., Boardman, R. P., Johnston, S. J., Scott, M., & O'Brien, N. S. (2014). Iridis-pi: A low-cost, compact demonstration cluster. *Cluster Computing*, *17*(2), 349–358. https://doi.org/10.1007/s10586-013-0282-7

Dampage, U., Bandaranayake, L., Wanasinghe, R., Kottahachchi, K., & Jayasanka, B. (2022). Forest fire detection system using wireless sensor networks and machine learning. *Scientific Reports*, *12*(1). https://doi.org/10.1038/s41598-021-03882-9

Dasari, P., Reddy, G. K. J., & Gudipalli, A. (2020). Forest fire detection using wireless sensor networks. *International Journal on Smart Sensing and Intelligent Systems*, *13*(1), 1–8. https://doi.org/10.21307/ijssis-2020-006

Diwedi, D. V., & Sharma, S. J. (2018). Development of a low cost cluster computer using raspberry Pi. *2018 IEEE Global Conference on Wireless Computing and Networking (GCWCN)* (pp. 11–15). IEEE. https://doi.org/10.1109/GCWCN.2018.8668647

Fernández-Cerero, D., Fernández-Rodríguez, J. Y., Álvarez-García, J. A., Soria-Morillo, L. M., & Fernández-Montes, A. (2019). Single-board-computer clusters for cloudlet computing in internet of things. *Sensors*, *19*(13), 3026. https://doi.org/10.3390/s19133026

Gupta, H. P., & Mishra, R. (2024). *Utilizing transfer learning and pre-trained models for effective forest fire detection: A case study of Uttarakhand* (Version 1). *ArXiv, abs/2410.06743*. https://doi.org/10.48550/ARXIV.2410.06743

Johnston, S. J., Basford, P. J., Perkins, C. S., Herry, H., Tso, F. P., Pezaros, D., Mullins, R. D., Yoneki, E., Cox, S. J., & Singer, J. (2018). Commodity single board computer clusters and their applications. *Future Generation Computer Systems*, *89*, 201–212. https://doi.org/10.1016/j.future.2018.06.048

Jones, M. W., Abatzoglou, J. T., Veraverbeke, S., Andela, N., Lasslop, G., Forkel, M., Smith, A. J. P., Burton, C., Betts, R. A., Van Der Werf, G. R., Sitch, S., Canadell, J. G., Santín, C., Kolden, C., Doerr, S. H., & Le Quéré, C. (2022). Global and regional trends and drivers of fire under climate change. *Reviews of Geophysics*, *60*(3). https://doi.org/10.1029/2020rg000726

Jones, M. W., Veraverbeke, S., Andela, N., Doerr, S. H., Kolden, C., Mataveli, G., Pettinari, M. L., Le Quéré, C., Rosan, T. M., Van Der Werf, G. R., Van Wees, D., & Abatzoglou, J. T. (2024). Global rise in forest fire emissions linked to climate change in the extratropics. *Science*, *386*(6719). https://doi.org/10.1126/science.adl5889

Kanani, P., & Padole, M. (2020). Analyzing ECG waves in fog computing environment using Raspberry Pi cluster. *2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)* (pp. 1165–1172). IEEE. https://doi.org/10.1109/I-SMAC49090.2020.9243398

Kang, Y., Jang, E., Im, J., & Kwon, C. (2022). A deep learning model using geostationary satellite data for forest fire detection with reduced detection latency. *GIScience & Remote Sensing*, *59*(1), 2019–2035. https://doi.org/10.1080/15481603.2022.2143872

Khan, A., Hassan, B., Khan, S., Ahmed, R., & Abuassba, A. (2022). DeepFire: A novel dataset and deep transfer learning benchmark for forest fire detection. *Mobile Information Systems*, *2022*, 1–14. https://doi.org/10.1155/2022/5358359

Khan, M. N. A., Tanveer, T., Khurshid, K., Zaki, H., & Zaidi, S. S. I. (2019). Fire detection system using raspberry Pi. *2019 International Conference on Information Science and Communication Technology (ICISCT)*, 1–6. IEEE. https://doi.org/10.1109/CISCT.2019.8777414

Marković, D., Vujičić, D., Mitrović, D., & Ranđić, S. (2019). Image processing on Raspberry Pi cluster. *International Journal Of Electrical Engineering And Computing*, *2*(2). https://doi.org/10.7251/ijeec1802083m

Newark Electronics. (n.d.). *RPI4-MODBP-4GB RASPBERRY-PI, SBC, Raspberry Pi4 B 4GB, BCM2711, ARM Cortex-A72, 4GB RAM, MicroSD, Linux, Wifi, 2x micro HDMI* Retrieved July 15, 2025 from https://www.newark.com/raspberry-pi/rpi4-modbp-4gb/rpi-computer-rpi-4-mod-b-4gb-1/dp/02AH3164

Raspberry Pi Foundation. (n.d.). *Raspberry Pi 4 Model B specifications*. Raspberry Pi. Retrieved July 14, 2025, from https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/

Reis, H. C., & Turk, V. (2023). Detection of forest fire using deep convolutional neural networks with transfer learning approach. *Applied Soft Computing*, *143*, 110362. https://doi.org/10.1016/j.asoc.2023.110362

Seydi, S. T., Saeidi, V., Kalantar, B., Ueda, N., & Halin, A. A. (2022). Fire-Net: A deep learning framework for active forest fire detection. *Journal of Sensors*, *2022*, 1–14. https://doi.org/10.1155/2022/8044390

Shamsoshoara, A., Afghah, F., Razi, A., Zheng, L., Fulé, P. Z., & Blasch, E. (2020). *Aerial imagery pile burn detection using deep learning: The FLAME dataset*. *ArXiv, abs/2012.14036*. https://arxiv.org/abs/2012.14036

Shamta, I., & Demir, B. E. (2024). Development of a deep learning-based surveillance system for forest fire detection and monitoring using UAV. *PLOS ONE*, *19*(3), e0299058. https://doi.org/10.1371/journal.pone.0299058

Singh, S. (2022). Forest fire emissions: A contribution to global climate change. *Frontiers in Forests and Global Change*, *5*. https://doi.org/10.3389/ffgc.2022.925480

Srinivas, P., D, U. M., Jha, S. K., Bajpai, I., & S, L. K. (2024). Smart early fire and smoke detection with transfer learning. *2024 IEEE Region 10 Symposium (TENSYMP)* (pp. 1–10). IEEE. https://doi.org/10.1109/TENSYMP61132.2024.10752273

US Army Engineer Research and Development Center. (n.d.). *Mpi4py/demo/compute-pi/cpi-cco.py at master erdc/mpi4py*. GitHub. Retrieved July 15, 2025, from https://github.com/erdc/mpi4py/blob/master/demo/compute-pi/cpi-cco.py

Yandouzi, M., Grari, M., Idrissi, I., Boukabous, M., Moussaoui, O., Azizi, M., Ghoumid, K., & Elmiad, A. K. (2022). Forest fires detection using deep transfer learning. *International Journal of Advanced Computer Science and Applications*, *13*(8). https://doi.org/10.14569/ijacsa.2022.0130832

Yang, H., Wang, J., & Wang, J. (2023). Efficient detection of forest fire smoke in UAV aerial imagery based on an improved Yolov5 model and transfer learning. *Remote Sensing*, *15*(23), 5527. https://doi.org/10.3390/rs15235527

Yunusov, N., Islam, B. M. S., Abdusalomov, A., & Kim, W. (2024). Robust forest fire detection method for surveillance systems based on you only look once version 8 and transfer learning approaches. *Processes*, *12*(5), 1039. https://doi.org/10.3390/pr12051039

Zhang, L., Wang, M., Fu, Y., & Ding, Y. (2022). A forest fire recognition method using UAV images based on transfer learning. *Forests*, *13*(7), 975. https://doi.org/10.3390/f13070975

Zhang, Q., Xu, J., Xu, L., & Guo, H. (2016). Deep convolutional neural networks for forest fire detection. *2016 International Forum on Management, Education and Information Technology Application* (pp. 568-575). Atlantis Press. https://doi.org/10.2991/ifmeita-16.2016.105

Zheng, B., Ciais, P., Chevallier, F., Yang, H., Canadell, J. G., Chen, Y., Van Der Velde, I. R., Aben, I., Chuvieco, E., Davis, S. J., Deeter, M., Hong, C., Kong, Y., Li, H., Li, H., Lin, X., He, K., & Zhang, Q. (2023). Record-high CO_2 emissions from boreal fires in 2021. *Science*, *379*(6635), 912–917. https://doi.org/10.1126/science.ade0805