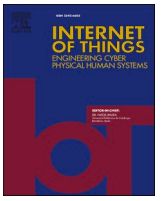
Internet of Things 25 (2024) 101013

Contents lists available at ScienceDirect

Internet of Things

journal homepage: www.sciencedirect.com/journal/internet-of-things



Sustainable collaboration: Federated learning for environmentally conscious forest fire classification in Green Internet of Things (IoT)

Ali Akbar Siddique [a], Nada Alasbali [b], Maha Driss [c,d], Wadii Boulila [c,d], Mohammed S. Alshehri [e], Jawad Ahmad [f,*]

[a] Department of Telecommunication Engineering, Sir Syed University of Engineering and Technology, Karachi 75300, Pakistan
[b] Department of Information Systems, College of Computer Science, King Khalid University, Abha 61421, Saudi Arabia
[c] RIOTU Lab, Prince Sultan University, Riyadh 12435, Saudi Arabia
[d] RIADI Lab, University of Manouba, Manouba 2010, Tunisia
[e] Department of Computer Science, College of Computer Science and Information Systems, Najran University, Najran 61441, Saudi Arabia
[f] School of Computing, Engineering and the Built Environment, Edinburgh Napier University, Edinburgh EH105DT, UK

ARTICLE INFO

Keywords:
Forest Fire Classification
Internet of Things (IoT)
Federated Multiparty Party Computation (FMPC)
Decentralized Data
Distributed computing

ABSTRACT

Forests are an invaluable natural resource, playing a crucial role in the regulation of both local and global climate patterns. Additionally, they offer a plethora of benefits such as medicinal plants, food, and non-timber forest products. However, with the growing global population, the demand for forest resources has escalated, leading to a decline in their abundance. The reduction in forest density has detrimental impacts on global temperatures and raises the likelihood of forest fires. To address these challenges, this paper introduces a Federated Learning framework empowered by the Internet of Things (IoT). The proposed framework integrates with an Intelligent system, leveraging mounted cameras strategically positioned in highly vulnerable areas susceptible to forest fires. This integration enables the timely detection and monitoring of forest fire occurrences and plays its part in avoiding major catastrophes. The proposed framework incorporates the Federated Stochastic Gradient Descent (FedSGD) technique to aggregate the global model in the cloud. The dataset employed in this study comprises two classes: fire and non-fire images. This dataset is distributed among five nodes, allowing each node to independently train the model on their respective devices. Following the local training, the learned parameters are shared with the cloud for aggregation, ensuring a collective and comprehensive global model. The effectiveness of the proposed framework is assessed by comparing its performance metrics with the recent work. The proposed algorithm achieved an accuracy of 99.27 % and stands out by leveraging the concept of collaborative learning. This approach distributes the workload among nodes, relieving the server from excessive burden. Each node is empowered to obtain the best possible model for classification, even if it possesses limited data. This collaborative learning paradigm enhances the overall efficiency and effectiveness of the classification process, ensuring optimal results in scenarios where data availability may be constrained.

* Corresponding author.
E-mail addresses: Nalasbali@kku.edu.sa (N. Alasbali), mdriss@psu.edu.sa (M. Driss), J.Ahmad@napier.ac.uk (J. Ahmad).

https://doi.org/10.1016/j.iot.2023.101013

Available online 29 November 2023

1. Introduction

Forests have an irreplaceable function in our everyday lives, providing us with a diverse range of valuable resources essential for sustaining life on Earth. They play a significant role in cleansing the air by effectively absorbing carbon dioxide and emitting oxygen, thus ensuring a healthier environment for all living beings [1]. Forests serve as invaluable natural reservoirs of minerals and materials that are critical for various industries and manufacturing processes. They supply us with essential resources like timber, wood pulp used in paper production, medicinal plants, and numerous other raw materials that drive our economies and sustain human livelihoods [2]. By harnessing the abundant resources offered by forests, we fulfill our material requirements while fostering economic growth and advancement. Additionally, forests play a vital role in preserving air quality [3]. Through the process of photosynthesis, trees, and plants absorb carbon dioxide—a greenhouse gas responsible for global warming and convert it into oxygen [4]. This natural process of carbon sequestration helps alleviate the impacts of climate change by reducing the concentration of CO_2 in the atmosphere [5]. Forests enrich the air with oxygen, supporting the respiratory systems of humans, animals, and other organisms, thus contributing significantly to the overall well-being and health of our ecosystems. Forests are the shields against sandstorms and play it part in maintaining the ecological balance [6]. As our global population continues to expand, the need for resources follows suit, raising serious concerns [7]. The unregulated deforestation happening worldwide has contributed to an increase in the planet's temperature, a direct response to meeting the demands of our growing population [8]. This surge in temperature stands as the primary catalyst for the occurrence of devastating forest fires. There are parts of the forest where the rise in temperature is significant, and dry conditions persist, these regions are more susceptible to fire outbreaks [9]. The effects of these fires are devastating, and their occurrence poses a severe threat to the environment, wildlife, and the availability of natural resources. Trees in the forest possess a highly flammable sap in their branches which makes them vulnerable to fire and the density of coniferous trees clustered together in close proximity to one another accelerates the spread of the fire [10]. As a result, these destructive forest fires result in the loss of millions of acres of forested land annually, causing substantial economic losses and significant negative impacts on the environment [11].

The forest fire outbreak that occurred in Australia in the year 2020 was devastating. It caused significant losses in terms of wildlife, resources, and human lives. More than 14 million acres of forest land were ravaged by fire destroying the habitats of thousands of animals and 1500 homes of the people living close to the forest. It also consumed the lives of 23 individuals leaving behind a sad and heartbreaking aftermath [12]. Forest fires have also caused destruction in other countries. In the years 2018 and 2019, it struck California and Amazon rainforests destroying wildlife and consuming hundreds of acres of land. It is a concern among the nations that are affected by forest fires, that over 85 % of these incidents are directly linked with human activity and the remaining 15 % are due to a natural effect such as lightning or climatic changes [13]. These fires that are induced by human activity can easily be prevented by placing strict regulations and curtailing the activities that contribute to forest fires. This can be further proved by the emergence of the COVID-19 pandemic, due to which, all countries around the globe imposed partial or total lockdowns reducing human activities as well as the occurrence of forest fires [14–16]. Governments around the globe have emphasized the importance of integrating intelligent surveillance and fire prediction systems to safeguard against the threat of forest fire [17]. Accurate forest fire detection plays a vital part in mitigating the risks associated with it, but prompt notification further increases the chance for the firefighters to reach the affected area swiftly and reduce further risk of damage [18]. The conventional means of monitoring forest fire is tiresome and not feasible in real-time as the density and area of the forests are immense, one the other hand, it may be dangerous for people to venture deep into the forest to monitor the occurrence of forest fire because there is a possibility for them to be attacked by wild animals [19]. These are some of the limitations which compromise the reliability of conventional human monitoring systems [20]. In this regard, IoT has emerged as a revolutionary concept for such applications associated with smart surveillance, integrating technologies like sensors, cloud computing, cameras, and wireless sensor networks [21]. Within the applied framework, IoT-enabled technologies are capable of generating immense amounts of data that can be efficiently processed and analyzed in real-time using Artificial Intelligence (AI) tools [22]. The immense data collected by the servers has led to the emergence of applications based on computer vision, enabling intelligence surveillance for different applications [23].

Integration of machine learning and deep learning algorithms within the domain of computer vision has reshaped the field of classification and prediction. Deep learning algorithms have solved most real-world applications based on classifications and predictions [24]. It can also play an instrumental role in the field of intelligent surveillance. Researchers around the world have shown a lot of interest in the field of deep learning, particularly in Convolutional Neural Networks (CNN) [25]. This interest is the direct cause of two significant factors: the availability of mass storage devices capable of staging immense amounts of data for deep learning applications and the introduction of high-performance Graphical Processing Units (GPU) capable of meeting the required computational demands. Still to this point, the acquisition of a large-scale dataset for a specific application remains a crucial obstacle in the pursuit of developing robust and efficient real-world applications [26]. Deep neural networks (DNNs) offer valuable capabilities in facilitating the early detection of forest fires and facilitating timely communication of critical information to the relevant authorities for prompt intervention. Recent advancements have witnessed the proposal of various deep learning-based mechanisms for fire and smoke detection, yielding notable outcomes. Maintaining continuous surveillance of forest areas enables the timely identification of fire incidents, allows the respective departments to take immediate action, and prevents the fires from escalating into large-scale disasters.

Since the dataset is for training the model is not in abundance, the contribution of this research is:

- Employing a Federated learning methodology to facilitate collaborative learning across a decentralized dataset.
- Integrating localized cameras into a Green IoT scenario to enable real-time video stream capture for the purpose of detecting and classifying forest fires.

- Conducting a comparative analysis between the proposed federated framework and other state-of-the-art Deep learning algorithms to evaluate their performance.

The forest fire detection problem is hampered by a significant constraint related to the availability of suitable datasets for deep learning. Previous research efforts addressing wildfire detection have utilized datasets that include various types of fires such as urban fires, riots, indoor fires, fires in open fields, and industrial fires. However, these datasets do not accurately represent forest landscapes, which poses a challenge when applying them to real-world forest fire detection scenarios. Consequently, the performance of detection models trained on such datasets may be suboptimal. The primary objective in tackling the forest fire detection problem is to achieve precise classification while minimizing the occurrence of false alarms and improving overall accuracy. This entails developing models that are specifically trained to detect and classify fires in forest environments. By focusing on the unique characteristics and patterns of forest fires, detection systems can be better optimized to accurately identify and distinguish between actual forest fires and other types of fires or non-fire elements.

The importance of a dedicated forest fire dataset cannot be overstated, as it provides the necessary training samples to capture the nuances of forest fire behavior, smoke patterns, and relevant contextual information. By utilizing a dataset specifically designed for forest fire detection, machine learning models can be trained to recognize key features indicative of forest fires and enhance their ability to discriminate between forest fires and non-fire instances. Accurate classification of forest fires is crucial for effective early warning systems, timely response, and resource allocation in fire management efforts. By reducing false alarms and improving accuracy, detection systems can help minimize the risk of delayed or inadequate responses to actual forest fires, ultimately contributing to more efficient and targeted firefighting strategies. Fig. 1 demonstrates the IoT-enabling technologies and the fields used for the federated approach.

2. Literature review

Over the course of the last few years, researchers from all over the globe have carried out a wide variety of investigations and developed a large number of classification methods in order to detect fire, smoke, and in some instances both [27]. The authors proposed using Faster R-CNN to create artificial smoke images as the basis for a smoke detection approach. The proposed method included adding smoke or simulated smoke images to the backdrop image of a forest to produce synthetic visuals [28]. A method utilizing a multi-color space locality binary pattern integrated with artificial intelligence is also introduced to detect smoke and fire signatures. Researchers have also utilized Long Short-Term Memory (LSTM) and You Only Look Once (YOLO) to classify smoke in forests, in this algorithm a lightweight teacher-student LSTM is used that reduces the number of layers and enhances smoke detection [29]. In another approach, researchers looked at using fog computing in tandem with CNN to spot fires in photographs. Following this, we discuss the study into forest fire localization and categorization that has been offered in the literature [30]. Researchers have also employed large-scale YOLOv3 to classify smoke and fire in the given images. As another research suggested, CNN-based flame detection may be used to keep an eye on forest fires. The CSPDarkNet53 is replaced with MobileNetv3 in their proposed technique, dubbed YOLO-Edge, which is based on YOLOv4. While the YOLOv4 backbone network is utilized to pinpoint the location of a fire, feature extraction is handled by MobilNetv3 [31].

Analysis of the different state-of-the-art CNN-based models is also used to classify forest fires and achieve good accuracy. Deep Believe Network (DBN) is used by researchers to detect smoke and forest fire. Dimensionality reduction, feature extraction, and classification are all handled by DBN, which is a stacked Restricted Boltzmann Machine in their proposed approach [32]. In [33], Zhao et al. introduced Fire_Net, a deep CNN-based approach to classifying forest fires. A deeper 15-layer CNN is used in their proposed Fire_Net model for classification, which is based on an AlexNet 8-layer CNN model. To solve the issue of smoke classification caused by forest fires, the authors proposed a unique solution known as attention-enhanced bidirectional long short-term memory (Abi-LSTM). Within the framework that they have proposed, Inceptionv3 is utilized for the extraction of spatial features, Bi-LSTM is used for the

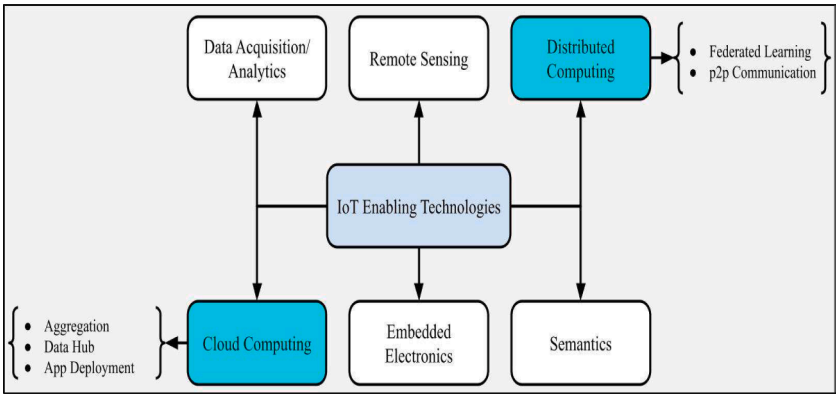


Fig. 1. IoT key enabling technologies.

extraction of temporal features, and the attention network is used to optimize the classification process [34]. A wildfire detection method employing transfer learning was introduced by Sousa et al. [35]. Their approach involved leveraging the pre-trained Inceptionv3 model's weights to classify images as either fire or non-fire images. In a particular research paper [36], the authors put forth a deep learning methodology for categorizing forest fires. Their approach involved utilizing the ForestResNet technique, which is built upon the ResNet50 model, to classify fire and smoke. In another investigation [37], a multilabel classification model was introduced for the detection of wildfires. The proposed method employed transfer learning with VGG16, ResNet50, and DenseNet121 models to classify various elements such as flames, smoke, non-fire instances, and other objects present in the images. Govil et al. [38] presented a terrestrial camera-driven approach for wildfire detection. Their method involved employing the Inceptionv3 model to classify images as either smoke or non-smoke, thus enabling the detection of wildfires. Sun et al. [39] put forth a CNN model for the classification of smoke in forest fires. Their proposed model incorporated batch normalization and multi-convolution kernels to optimize classification accuracy. In a separate study [40], the authors introduced a novel dataset called DeepFire, designed specifically for forest fire detection. The research explored the dataset's performance using different machine learning algorithms and presented a transfer learning solution based on the VGG19 model.

3. Improving performance through dataset acquisition, preprocessing, and augmentation

The inadequate representation of various characteristics or attributes in the training dataset is highly probable, leading to inaccurate reflections within the trained model. To address this issue, numerous pre-processing methods can be explored in order to enhance the dataset. These methods encompass a range of algorithms such as translation, scaling, rotation, and noise suppression, which can be employed to improve the dataset's quality and effectiveness [41]. The primary cause for the limited representation of characteristics or attributes in the training dataset lies in the insufficiency of information it contains. As a result, the trained model fails to capture the full complexity and diversity of the data it encounters during inference. However, this challenge can be overcome by implementing various pre-processing techniques to enhance the dataset's comprehensiveness [42]. The research utilized a dataset available on Kaggle, which was introduced by [43]. A few image samples of the dataset used are displayed in Fig. 2.

In the context of federated learning, when the availability of images is restricted, certain significant factors need to be considered. The limited availability of images can be a significant obstacle in the development of robust and precise models. Due to a limited number of data points available for learning, the model may have difficulties in capturing intricate patterns and exhibiting efficient generalization when presented with novel, unseen instances. This phenomenon raises the likelihood of overfitting, wherein the model excessively adapts to the peculiarities of the restricted dataset, which may result in suboptimal performance when applied to real-world data. Moreover, the process of federated learning may exhibit less stability and reliability, as the updates received from individual clients could be more susceptible to noise and outliers owing to the limited size of the sample. In order to address these problems, it is imperative to employ strategies such as data augmentation to optimize the utilization of the existing data and improve the efficacy of the federated learning system.

The dataset is uniformly scaled to maintain the resolution of 224 $\times$ 224. Every image in the dataset is adjusted to maintain the impartiality of individual pixels when analyzing the images. The coordinate values *(x, y)* describe the locations of pixels in the original image and are used to translate images. The coordinates are changed for the translated image by deducting the translation amounts *(dx, dy)* to get new coordinates *(x - dx, y - dy)*. The relevant pixel values for the translated image can be derived by replacing these translated coordinates into the original image. The image coordinate system typically starts in the top-left corner, expanding horizontally to the right and vertically downward. The number of pixels by which the image is moved along the x-axis and y-axis,

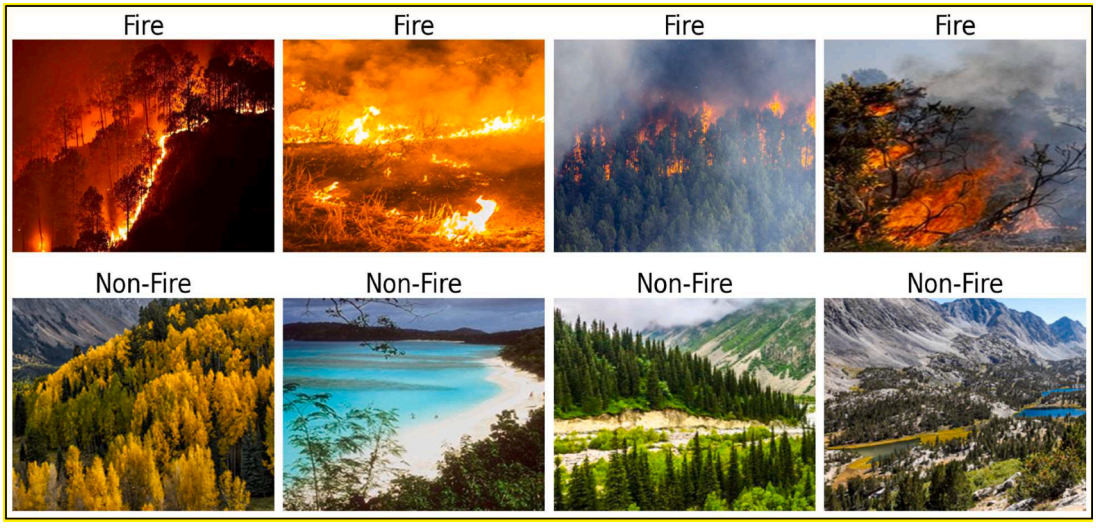


Fig. 2. Sample Dataset of Forest Fire Depicting Positive (Fire) images and Negative (Non-Fire) images.

respectively, is determined by the translation amounts, *dx*, and *dy*. A shift to the right is produced by positive dx values, whereas a shift to the left is produced by negative *dx* values. Similar to how negative *dy* values result in an upward shift, positive *dy* values cause a downward movement. As a result, there may be more variability in the data and a wider range of traits or qualities covered. In computer vision tasks, for instance, translating images may imitate changes in perspective, viewpoint, or position, allowing the model to develop more reliable representations. Eq. (1) depicts the method of Image Translation, where *T(x, y)* is an original image and *I(x - dx, y - dy)* represents a Translated Image. Fig. 3 represents the outcome of the translation when applied on the images.

$$T(x,\ y)\ =\ I(x\ -\ dx,\ y\ -\ dy) \tag{1}$$

The term "image scaling" is used to describe the process of increasing or decreasing the size of an image as given in Fig. 4. This approach is widely used in several disciplines, including Computer Graphics, Image Processing, Web development, and photography. There are several types of scaling techniques such as Nearest neighbor, Bilinear Interpolation, Bicubic Interpolation, etc. In the proposed work, Bilinear interpolation is used to augment the dataset. When scaling images, bilinear interpolation is often used. The method relies on a weighted average of the values of the four neighboring pixels in the original image to estimate the new pixel values. This method depends on the assumption that surrounding pixels have linearly varying brightness. Bilinear interpolation, in contrast to the more straightforward closest neighbor interpolation, produces smoother results when upscaling an image. Assume an original Image *T* with the dimensions of *(x, y)* and the targeted image *I'* with the dimensions of *(x', y')*. For the purpose of upscaling $x' > x$ *and* $y' > y$ and for downscaling the scenario is reversed. In order to identify the value of pixel *T'(x', y')* in a new reconstructed image, bilinear interpolation is employed using four adjacent pixels. The pixel coordinates in the original image corresponding to *T'(x', y')* can be calculated using scaling factors S_X and S_Y, which can be seen in Eqs. (2) and (3). Within the domain of federated learning, the utilization of bilinear interpolation proves to be a significant asset in the estimation of pixel values in situations where a given device lacks access to specific images. This approach functions by calculating the weighted average of the four nearest known pixels, so enabling a smooth transition between them. This feature is especially beneficial in tasks that prioritize the preservation of spatial relationships within images, such as object recognition or segmentation. In contrast to the nearest neighbor interpolation technique, which has the potential to generate blocky artifacts, bilinear interpolation provides a more seamless transition, leading to visually enhanced and more lifelike images, particularly when enlarging low-resolution data. Therefore, within the context of federated learning, it can be argued that bilinear interpolation is a more advantageous option compared to closest-neighbor interpolation. This assertion is based on the fact that bilinear interpolation has the ability to produce images of superior quality, characterized by improved spatial coherence.

$$x = x^{'}/S_X \tag{2}$$

$$y = y^{'}/S_Y \tag{3}$$

Next, to find the four surrounding pixels in the original image *T* based on the integer coordinates *(x, y)*. Let P_1 be the pixel at *[floor (x), floor(y)]*, P_2 at *[ceil(x), floor(y)]*, P_3 at *[floor(x), ceil(y)]*, and P_4 *at [ceil(x), ceil(y)]*. Fractional parts X_f and Y_f are calculated which point to the fact that how far the targeted point *(x', y')* is from the top-left pixel of the four nearest pixels. Eqs. (4) and (5) demonstrate the calculation of the fractional parts. After applying Bilinear interpolation in Eq. (6), we acquire *T'(x', y')*.

$$x_f\ =\ x - floor(x) \tag{4}$$

$$x_f\ =\ y - floor(y) \tag{5}$$

$$T^{'}(x^{'},\ y^{'}) = \left(1 - X_f\right) * \left(1 - Y_f\right) * T[floor(x),\ floor(y)] + X_f * \left(1 - Y_f\right) *\ T[ceil(x),\ floor(y)] + \left(1 - X_f\right) *\ Y_f *\ T[floor(x),\ ceil(y)] + X_f * Y_f *\ T[ceil(x),\ ceil(y)] \tag{6}$$

In order to accomplish rotational modification, an image goes through an affine transformation, as shown in Eqs. (7) and (8) for the Cartesian coordinate system. Assume that the variable *'θ'* will denote the degree of rotation in this case. A rotated image may be produced by first applying the specified transformation matrix to each pixel of an image as shown in Fig. 5, and then altering the value

Fig. 3. (a) Original image 1. (b) Translated image 1. (c) Original image 2. (d) Translated image 2.

Fig. 4. (a) Original image 1. (b) Upscaled image 1. (c) Original image 1. (d) Downscaled image 2.

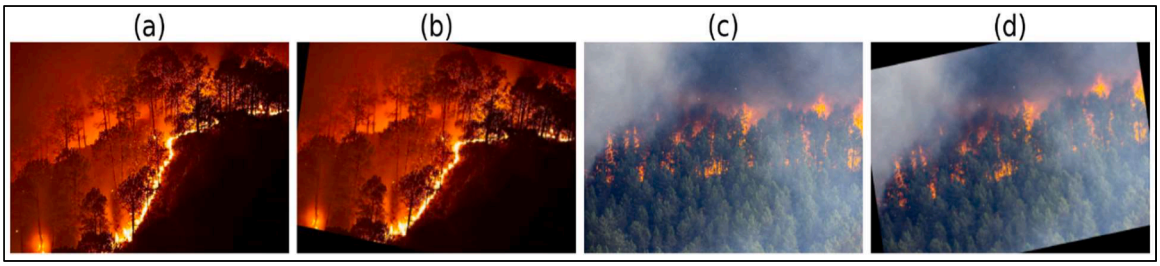


Fig. 5. (a) Original image 1. (b) Right rotated image 1. (c) Original image 2. (d) Left rotated image 2.

of the parameter denoted by the letter 'θ' in accordance with this change. Since, $(x,\ y)$ are the coordinates of a single pixel, $(x_c,\ y_c)$ will the center (pivot) rotation coordinates. $(x_r,\ y_r)$ represents the pixel coordinates of the rotated image.

$$x_r = \cos(\theta) * (x - x_c) - \sin(\theta) * (y - y_c) + x_c \tag{7}$$

$$y_r = \sin(\theta) * (x - x_c) - \cos(\theta) * (y - y_c) + y_c \tag{8}$$

There are 1520 total images in the collection sourced from Kaggle for the proposed federated algorithm. There are 760 of these that focus on fire and depict different fires or fire-related situations. The other 760 images, which depict various settings and things devoid of any reference to fire, are non-fire images. The number of images based on fire and those not based on fire are equal. For classification tasks, having a balanced distribution is essential since it guards against biases and guarantees that the model doesn't learn to prefer one class over another. However, there is a need for further data augmentation in the context of utilizing this dataset for federated learning, which is a decentralized machine learning strategy where data stays on the local devices and is only aggregated for model training. A typical machine learning method for artificially increasing the size of a dataset is called data augmentation. New images may be created by applying different alterations to the already-existing images. Although these enhanced images are not exact replicas of the originals, they nonetheless communicate the same underlying ideas. The number of images accessible for federated learning may be greatly expanded by expanding the dataset. This is advantageous since federated learning depends on local data from several devices or places, and having a large centralized dataset is sometimes difficult. Each device or location may contribute additional data by creating augmented images, creating a richer and more varied dataset for the federated learning model to be trained on. Due to data augmentation, the dataset size has grown, which may improve federated learning performance. A bigger, more diverse dataset enables the model to capture a greater variety of patterns and characteristics, which may enhance generalization and accuracy when working with fresh, untested data. Table 1 represents the datasets with and without augmentation. When confronted with networks of considerably greater scales, such as those comprising more than 100 nodes, it becomes imperative to adopt approaches that can effectively address the heightened intricacy. First and foremost, it is imperative to guarantee that the algorithms utilized possess scalability, enabling them to effectively handle a larger quantity of nodes without incurring an excessive rise in processing requirements. Furthermore, it is important to consider the memory and storage demands, which may need the utilization of more advanced hardware or cloud-based resources. The utilization of parallel processing algorithms can efficiently allocate the computing workload among numerous cores or machines, resulting in an accelerated analytical process. In addition, the utilization of optimal data structures for storing and processing can assist in addressing concerns related to time complexity. These precautionary measures

Table 1
Pre- and post-augmented dataset.

S. No.	Pre-augmentation dataset		Post-augmented dataset	
	Class	Quantity	Class	Quantity
1	Fire	760	Fire	2500
2	Non-fire	760	Non-fire	2500

jointly contribute to a comprehensive methodology for managing networks of significant scale.

4. Experimental setup

Five edge nodes were set up and tuned for smooth communication with the central server, which held the initial base model, to apply the proposed federated learning approach. These edge nodes served as the endpoints for the safe and private processing of data coming from various devices within the network. In this configuration, the edge nodes served as participants in the learning process while the central server served as its coordinator. The central server would receive requests from each edge node seeking model changes, allowing a distributed and cooperative learning strategy. This strategy is advantageous because it enables the model to draw knowledge from a wide variety of data sources while preserving the localization and security of sensitive data on the edge nodes. The total system profited from the combined knowledge of all edge nodes by using this federated learning strategy without compromising user privacy or the need to centralize data. The model's performance and flexibility to different real-world circumstances were improved via the distributed learning process.

4.1. Stochastic data allocation for edge processing

Edge computing has emerged as a promising paradigm to address the increasing demand for real-time data processing and reduced latency in various applications. In traditional centralized systems, the burden of data processing and analysis lies solely on the central server, leading to potential bottlenecks and increased response times. To alleviate these issues, a distributed approach is necessary aiming to optimize data allocation among edge devices. This method entails a non-deterministic data assignment to edge nodes, allowing for a more efficient and balanced data processing framework. The core idea of Stochastic Data Distribution is to distribute data across multiple edge nodes using a probabilistic approach. Rather than following deterministic allocation methods, where data is assigned based on fixed rules, a stochastic approach randomly allocates data to the edge nodes. This randomness introduces diversity in the data processed by each edge node, leading to a more comprehensive understanding of the underlying data distribution. Devices within the network have the possibility to possess distinct and finite computational resources. So, in order to make the training process more efficient, we divide the dataset into batches and then assign each edge device a particular batch to handle.

In contrast, the simplest method of splitting the information into batches and distributing it to edge devices may not be the most effective one. One edge node can have less complicated data than the others, which might result in an uneven workload allocation and poor performance. Using a stochastic data distribution strategy, each training cycle randomly shuffles and divides the dataset into batches. The edge nodes are then randomly allocated to the batches in an effort to ensure a more even distribution of data complexity across all devices. By doing this, the training process is improved, and the edge nodes are better able to cooperate to improve performance as a whole. The possible bias that can develop when batches are allocated in an intuitive manner is addressed by this method. It assures each device gets data with a range of characteristics that are roughly comparable by using the stochastic data distribution approach. This contributes to a more equitable training environment on all edge nodes. Verifying the datasets of every edge device in the network is essential in the context of federated learning. A damaged dataset may significantly impede the local model training process on any node. The model's performance would suffer as a consequence of learning from false or inaccurate data. In federated learning, a global model is created by combining model updates from many nodes. The global model's overall quality may suffer if a node with a damaged dataset participates in training and its updates are included in the aggregate. Corrupted data may cause biases, outliers, or inconsistencies that have an influence on learning and ultimately the performance of the final model. In order to get reliable and accurate results, it is crucial to ensure the data's integrity throughout all edge devices. A probability distribution function (PDF), which expresses the likelihood of certain values happening, can be used to represent the stochastic data. The PDF will be a Gaussian function, for instance, if the data have a normal distribution. The likelihood that a stochastic variable will have a value that is lower than or equal to a certain threshold is represented by the cumulative distribution function (CDF), which is the integral of the PDF. Normal distribution is a continuous probability distribution that is symmetric and bell-shaped. It is described by two parameters: the mean (μ) and the standard deviation.

The PDF of the Gaussian distribution is given by Eq. (9). Keeping in mind that the chance of any particular value of x is infinitesimally small since the Gaussian distribution is a continuous distribution. Probabilities, on the other hand, are often computed across intervals or ranges of values. You would integrate the PDF $f(x)$ from a to b in order to get the probability of an interval [a, b]. CDF is given in Eq. (10).

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2} \tag{9}$$

Table 2
Decentralized data distribution among 5 distinct nodes.

No	Class	Node (1)	Node (2)	Node (3)	Node (4)	Node (5)
1	Fire	350	525	800	375	425
2	Non-Fire	350	525	800	375	425

$$F(X) = \int_{-\infty}^{x} f(x)dx \tag{10}$$

The choice of the edge nodes is influenced by a number of variables, including proximity to the data source, communications abilities, and resource availability. The group of edge devices existing in the network is represented by Eq. (11) where *i* is the number of devices. Eq. (12) is used to determine the likelihood that edge node *Ni* will be chosen to distribute the data. In Table 2, a dataset with five nodes is demonstrated, each of which is classified as either "Fire" or "Non-Fire." Each column in the table represents a separate node, and each row in the table represents a class. The data samples connected to each class at each node are indicated by the numbers in the table. Table 3 presents the distribution of data samples across different nodes for a classification task involving fire and non-fire detection. Each node represents a subset of the dataset. For instance, in Node (1), during the training phase, there are 298 samples classified as Fire and an equal number of 298 samples classified as Non-Fire. In the testing phase, there are 52 samples each for both Fire and Non-Fire classes. This pattern continues for other nodes as well. For example, in Node (2), there are 446 samples each for both Fire and Non-Fire during training, and 79 samples each for testing. The same distribution trend applies to Nodes (3), (4), and (5). In order to validate the accuracy of the Global Model, 250 images are used, out of which 143 are non-fire images and 107 are fire images.

$$N = \{N_1, N_2, N_3 \ldots\ldots N_i\} \tag{11}$$

$$P(N_i) = \frac{w(N_i)}{\sum_{j=1}^{n} w(N_j)} \tag{12}$$

4.2. 3-Layered basic CNN architecture

Training a model through federated learning involves utilizing multiple devices, each with unique data distributions, processing capabilities, and network connections. The choice of an ideal model becomes crucial, one that balances flexibility and simplicity. Such a model should accommodate data heterogeneity while also exhibiting strong generalization capabilities for unseen data. Opting for a flexible model with fewer parameters can mitigate overfitting risks in resource-limited scenarios, leading to reduced computational and communication requirements. Additionally, a simpler model offers enhanced readability, a critical factor in domains like healthcare and finance where model clarity and reliability are paramount. Although a simpler model may exhibit slightly lower accuracy compared to a more complex counterpart, such differences might not be statistically significant in many cases. Therefore, deploying a simpler model with fewer parameters and lower computational and memory demands proves practical and suitable for resource-constrained devices, especially in federated learning contexts.

The design of the neural network is composed of numerous layers, each of which processes input pictures with a resolution of 224 × 224 pixels and three-color channels (RGB). The first layer is known as the input layer, and it does not include any trainable parameters. Following this, the Conv2D layer, which consists of 32 filters with a size of 3 × 3, applies convolution to the input. As a consequence of the spatial dimension reduction, the output has the form of (222 × 222 × 32). There are 896 parameters for this layer. After applying the succeeding MaxPooling2D layer, the spatial dimensions are cut in half, which results in an output shape with the coordinates (111 × 111 × 32). After that, there is one more Conv2D layer in the network, and this one includes 64 filters of size 3 × 3. This further decreases the spatial dimensions, which ultimately results in an output shape of (109 × 109 × 64). The total number of attainable values for all of these parameters is 18,496. After that comes another layer of MaxPooling2D, which results in the output shape being (54 × 54 × 64). After convolution, the output form will be (52 × 52 × 64) since the third Conv2D layer has 64 filters with a size of 3 × 3. The number of parameters for this layer is 36,928. The spatial dimensions are cut down even more by a later MaxPooling2D layer, which results in an output shape with the coordinates (26 × 26 × 64). The output from the layer below is brought into the Flatten layer, where it is transformed into a one-dimensional vector with the shape (43,264,). This layer does not have any trainable parameters at this time. After that, a Dense layer consisting of 512 neurons is added, and as a result, an output shape of (512,) is generated. There are 22,151,680 parameters associated with this layer. In the very top Dense layer, also known as the output layer, there are two neurons for binary classification, which means there are two different categories. (2) is the output shape, and there are 1026 parameters included inside this layer. Table 4 represents the 3-layered CNN model and Fig. 6 demonstrates its flow for input and output image shape.

Table 3
Training and testing dataset for 5 distributed nodes.

		Fire	Non-Fire
Node (1)	Training	298	298
	Testing	52	52
Node (2)	Training	446	446
	Testing	79	79
Node (3)	Training	680	680
	Testing	120	120
Node (4)	Training	319	319
	Testing	56	56
Node (5)	Training	361	361
	Testing	64	64

Table 4
3-layered classification model employed for Federated Learning.

Layer type	Output shape	Number of parameters
Input layer	(224 × 224 × 3)	0
Conv2D (32 filters)	(222 × 222 × 32)	896
Dropout (20 %)	(222 × 222 × 32)	0
MaxPooling2D (2 × 2)	(111 × 111 × 32)	0
Conv2D (64 filters)	(109 × 109 × 64)	18,496
Dropout (20 %)	(109 × 109 × 64)	0
MaxPooling2D (2 × 2)	(54 × 54 × 64)	0
Conv2D (64 filters)	(52 × 52 × 64)	36,928
Dropout (20 %)	(52 × 52 × 64)	0
MaxPooling2D (2 × 2)	(26 × 26 × 64)	0
Flatten	(43,264,)	0
Dense (256 neurons)	(512,)	22,151,680
Dense (Output)	(2)	1026

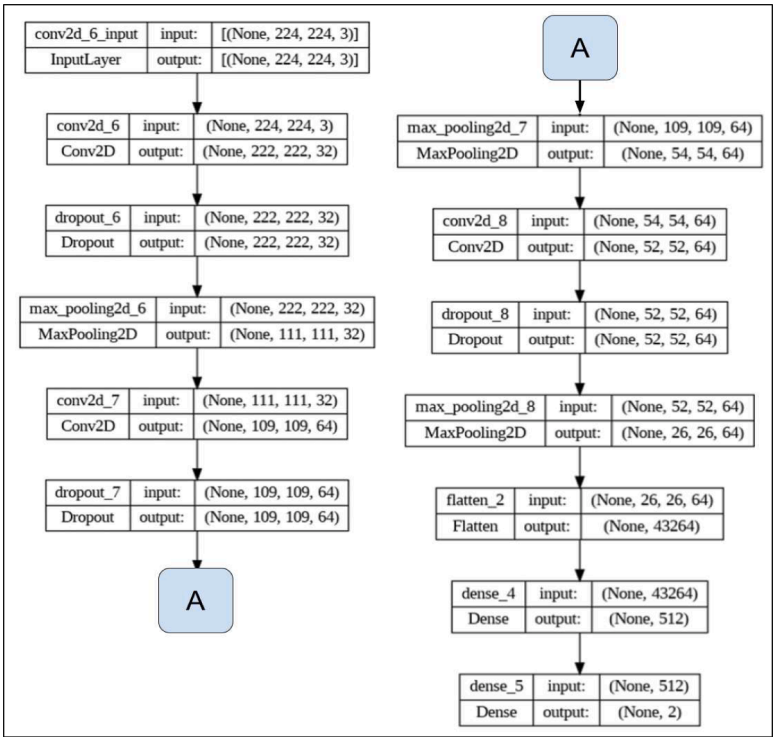


Fig. 6. Flowchart of the 3-layered CNN model used as a base model for Federated Learning.

One noticeable advantage that arises in a federated learning setting is when a device is devoid of an entire category or class of data or even has less data as compared with other nodes. The rationale behind the implementation of the aggregation process is basically intended to tackle such situations. The central server has the ability to collect and combine updates from all devices involved in the aggregation process, including those that may not have certain classes of updates. This implies that the global model, which is located on the central server, integrates the pooled knowledge from all devices, so successfully compensating for any missing or low data categories. As a result, the parameters of the model are iteratively optimized and modified to account for the missing class, enabling it to generate accurate predictions or classifications even when data from specific devices is unavailable. The adaptability and resilience of federated learning are enhanced by this particular trait, which allows for the effective handling of fluctuations in the data accessible on participating devices.

Sigmoid and rectified linear unit (ReLU) activation functions are used in this research. These functions were selected in order to improve the process of optimization and to inject non-linearity into the behavior of the model. Producing the elementwise maximum between 0 and the input value, which is denoted as $\max(0, z)$, is one of the ways that the ReLU activation contributes to the process of learning complex features. The sigmoid function, which is also known as the logistic function, was applied in the process of recognizing forest fires as the work at hand. This function, which returns a single result that might range from 0 to 1, provides the likelihood that a forecast will be correct. Eqs. (13) and (14) represent the Sigmoid and ReLU functions respectively.

$$Sigmoid_z = \frac{1}{1+e^{-z}} \tag{13}$$

$$ReLU = \max(0, z) \tag{14}$$

4.3. Hyperparameters selection for model training

The goal of the Federated Averaging (FedAvg) optimizer, which is used extensively in federated learning, is to train a global model while simultaneously maintaining data decentralized. Multiple communication loops are required during the procedure. At the beginning of each new round, a subset of the customers who are eligible to participate is chosen. For each selected client i, it downloads the current global model parameters θ_i, computes the local gradient $\nabla L_i(\theta_G)$ based on its local loss function $L_i(\theta)$, and updates its local model parameters using a local optimizer as depicted in Eqs. (15) and (16). μ is a learning rate which is selected to be 0.001 for the proposed federated scheme.

$$\theta_{i(new)} = LocalOptimizer(\theta_G,\ \nabla L_i(\theta_G)) \tag{15}$$

$$\theta_{i(new)} = \theta_G - \mu \nabla L_i(\theta_G) \tag{16}$$

The next stage in federated learning is to aggregate these changes on the central server to acquire the updated (θ_G) for the next communication cycle. During the local training phase, each client updates its model based on its own local dataset. In federated learning, this aggregation procedure is essential because it promotes collaborative learning while protecting the privacy of individual clients' data. The clients send their revised model parameters back to the central server after completing their local training. Multiple clients send these model changes to the server, but not every client may be chosen to actively take part in a particular communication cycle. The number of chosen customers is indicated by the N, and the selection method may be based on random sampling. The server conducts a weighted average of the model updates received from the chosen clients in order to acquire the new global model parameters $\theta_{G(new)}$ for the next round in Eq. (17). This weighted average makes sure that each client's contribution is proportional to the quantity of updates the client has provided. $\theta_{G(new)}$ represents the updated model parameters received from client i. The summation depicts the process of iteration over all the selected clients in the current round. The division by N normalizes the summation, ensuring that the average model is influenced equally by all selected clients. m_i is the number of data samples at client i. Consequently, the global model precisely represents the information and insights gained from all the involved customers. Given that the client datasets may be diverse and non-IID (non-independent and identically distributed), weighted averaging is crucial in federated learning. For the global model, certain customers can have data that is more relevant or representative than others. The federated learning method reduces the effects of data imbalances between clients and prevents overfitting to any one client's data by aggregating the updates with weights proportionate to the number of updates received.

$$\theta_{G(new)} = \frac{1}{N}\sum_{i=0}^{N}\left(m_i \times \theta_{i(new)}\right) \tag{17}$$

4.4. Validation of the proposed federated framework via performance metrics

When evaluating the effectiveness of a machine learning model, accuracy is one of the measures that is used the majority of the time. It determines what percentage of the model's overall predictions are accurate and reports the results as depicted in Eq. (18). To determine how accurate a prediction is, just take the number of right forecasts and divide it by the total number of predictions. Here, TP stands for True Positive, TN is Trude Negative, FP is False Positive and FN is False Negative.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{18}$$

Precision is a statistic that represents the percentage of genuine positive predictions out of all the positive forecasts generated by the model as given in Eq. (19). In other words, precision measures how accurate a model's positive predictions are. It evaluates how effectively the model steers clear of producing false positive results.

$$Precision = \frac{TP}{TP + FP} \tag{19}$$

The percentage of accurate positive predictions provided by the model relative to the total number of actual positive cases included in the dataset is what is meant by the term "recall" as depicted in Eq. (20). It determines how well the model steers clear of producing false negatives.

$$Recall \Big/ Sensitivity = \frac{TP}{TP + FN} \tag{20}$$

The F1 score is a single metric that balances both the accuracy and recall measurements, and it is calculated by taking the harmonic mean of these two metrics and its expression is given in Eq. (21). It produces a single score that indicates the performance of the model in terms of both false positives and false negatives. When there is a significant imbalance in the number of positive and negative

examples present in the dataset, the F1 score might be beneficial.

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{21}$$

5. Proposed approach for facilitating collaborative learning via IoT-integrated Federated Multi-Party Computation (FMPC)

Federated Learning is a form of decentralized learning technique that utilizes data stored on the edge devices linked to the server to train the model. By reducing the expenses of data transit and centralized storage, this method preserves the data's security. The final product is a global model that has been trained with these scattered bits of data and is applicable to any edge device that is interconnected within the application. While datasets are kept on a single server by traditional machine learning methods. In circumstances involving continuous learning, this might lead to issues about scalability as well as higher communication costs. One of Federated Learning's main advantages is its ability to overcome network latency problems. Since mobile computing applications are expected to have short response times, this capability is very crucial. Because model training is done locally on edge devices, federated learning eliminates the requirement for a central server. This opens the way for edge node-based real-time model training. It's also important to remember that traditional machine learning techniques need edge nodes linked to a central server to communicate their unique data. This information could be private and include delicate information like medical records. Additionally, regular communication between the edge nodes and the server is required for conventional machine learning methods. In contrast, Federated Learning's capacity to solve these privacy issues is what distinguishes it as innovative. This is accomplished by strategically limiting the amount of data shared between the central server and edge nodes. A network of edge nodes, a central server, and an advanced learning algorithm are the three key elements that support the federated learning framework. Fig. 7 demonstrates the procedure of the federated learning process.

The Internet of Things (IoT) plays a significant part in Industry 5.0, which is characterized by the convergence of digital and physical technologies to transform many businesses. For instance, Internet of Things technology helps link devices and exchange data for improved decision-making in the area of fire categorization. This is significant since it helps enhance the precision with which fires can be identified. In this configuration, multiple devices are able to effortlessly collaborate with one another and safely communicate information concerning fire events. This collaboration results in improved methods for detecting and responding to fires in real time. When it comes to classifying fires using IoT, the number of connected devices directly correlates to the amount of communication that must take place between them. Each device transmits its data about fires to a central hub, which then compiles all of the information, improves the insights that are sent back to the devices, and repeats this process. However, if there are a large number of devices, this communication may become sluggish and result in delays. This is the point at which things start to get more complicated. Even though the system is meant to deal with a large number of devices, there are instances when it moves at a snail's pace. To keep up with everything, the central node that is responsible for data collection and processing needs to put in additional effort. Therefore, although the Internet of Things (IoT) is fantastic for making things smarter, particularly for fire classification, we need to make sure that the system can manage all of the information without becoming too sluggish when there are too many devices. It is comparable to ensuring

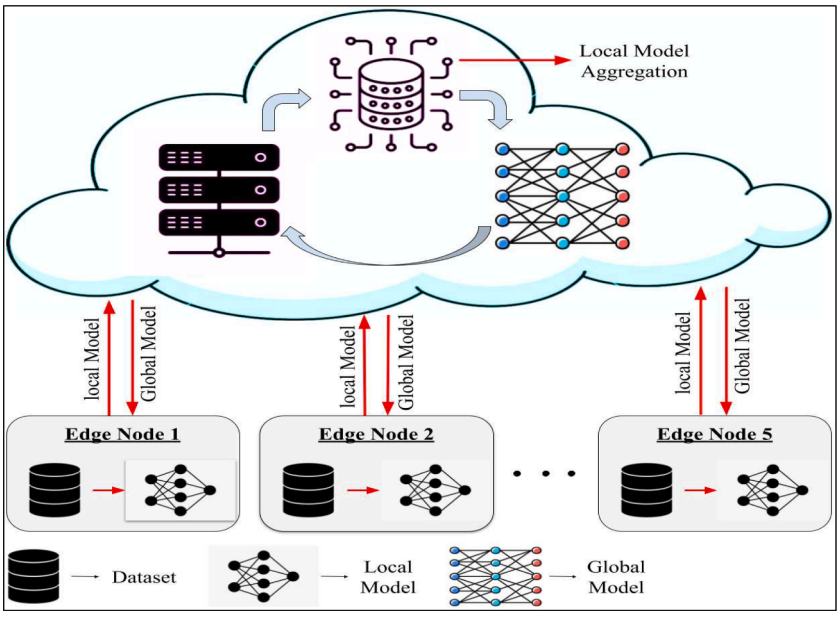


Fig. 7. Federated learning in IoT integrated framework.

that a team functions well together even when there are a large number of players on the field. Edge computing is a computer paradigm that entails the processing of data in close proximity to its origin or point of utilization, as opposed to relying exclusively on centralized cloud servers. Computational capabilities are extended to devices or "edges" within a network, including Internet of Things (IoT) devices, smartphones, or local servers. This methodology presents numerous benefits, such as decreased latency, greater bandwidth efficiency, and heightened privacy and security through the proximity of critical data to its point of origin. Edge computing plays a vital role in applications that necessitate immediate or nearly immediate processing, such as autonomous vehicles, smart cities, and industrial automation. The distribution of computational workloads over a network, known as edge computing, facilitates enhanced efficiency and responsiveness inside systems. Consequently, this approach contributes to the development of more resilient and dependable applications across several domains. A concept of utilization of Edge computing in the proposed word is provided in Fig. 7.

Nodes in the area of federated learning are often spread across several platforms or locations. These nodes keep a large range of data as a consequence of this dispersion, leading to some degree of statistical diversity. As the number of nodes in the network increases, this variety and heterogeneity in data distribution might become more pronounced. This specific aspect has the ability to affect how the whole global model performs. This is owing to the fact that it is necessary to balance out the varied contributions made by each node while the information from each one is being gathered. There is a chance that certain nodes may have data that is less representative of the whole sample or of poorer general quality. Each node that participates in the process trains its own model using a unique set of data in the federated learning approach. The number of nodes in the network closely correlates to the rising demand for computing resources. The central server or aggregator also faces additional demands as it manages the flow of model updates and subsequent aggregations. Each node requires more resources for training and generating predictions. Federated learning is often used to keep data private and secure since the actual training data remains on these local nodes. However, as nodes increase, it raises a fresh set of concerns around security and privacy. The difficulties will increase as the nodes continue to grow. In a system with more nodes, there may be more weak points, which might make the system more vulnerable. Additionally, when a network's nodes expand, the risks associated with privacy violations or inference attacks where sensitive data is inferred from seemingly innocuous data might also rise. Federated learning will need to navigate the complicated dynamics of balancing multiple inputs, optimizing resource allocation, and fortification itself against the risk of potential security problems as it absorbs an expanding number of nodes to increase both privacy and collaboration.

5.1. Local model training

During each and every stage of the training process, the loss function is of the utmost significance when it comes to deep learning, which makes use of neural networks that have very complex topologies. Deep neural networks are made up of a great number of stacked, linked layers; each of these layers contributes to the ability of the model to grasp sophisticated data-based interactions. In order to train these networks, it is necessary to iteratively alter the millions of parameters that are responsible for governing the behavior of the neural network. The basic goal of training is to get a loss function that is as little as possible. This loss function acts as a gage for how closely the model's predictions match the actual target values in the training dataset. Eq. (22) represents the generalize Loss function. The selection of the suitable loss function is not random; rather, it is inextricably linked to the characteristics of the objective that the model seeks to realize. Loss functions must vary depending on the kind of work being performed. Here, $\mathscr{L}(\theta)$ is a loss function, θ represents the local model parameters, N is the number of training samples, ℓ is per sample loss function, $f_\theta(x_i)$ is the model's prediction for input x_i, and y_i is the true target value for input x_i.

$$\mathscr{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N} \ell(f_\theta(x_i), y_i) \tag{22}$$

In the process of optimizing the performance of a model, two essential methods emerge as front and center: the gradient descent update and regularization. These components serve as the foundation of efficient model optimization, and they play a critical part in determining how a model's capabilities and generalization are shaped. The gradient descent update is a basic method that is used to fine-tune a model's parameters while it is being trained. As such, it is one of the most important aspects of model improvement. Throughout the course of the training procedure, the model will work toward minimizing the loss function, which measures the degree to which predictions deviate from actual values. The model will make adjustments to its parameters in a way that goes against the gradient of the loss function so that it can make its way across the terrain of optimization. This computed alteration directs the trajectory of the model toward areas with parameter values that generate lesser loss, which ultimately fosters greater prediction accuracy.

The capacity of a model to successfully generalize is significantly improved when regularization is included into the optimization landscape as given in Eq. (23). Here, N symbolizes the total number of model parameters, and λ is the regularization strength. Overfitting is a typical issue that emerges in the field of machine learning. This problem occurs when models capture noise rather than underlying patterns, which leads to poor performance on data that has not yet been seen. The role of regularization is similar to that of a watchman; it maintains harmony between complexity and generalization. One of the most common variations, known as L2 regularization, involves including a penalty term into the loss function that is proportional to the extent to which the model's parameters vary. This results in a tendency for the parameters to adopt lower values, which helps to reduce excessive volatility and promotes a more generic model.

$$\mathscr{L}_{reg}(\theta) = \frac{\lambda}{2}\sum_{j=1}^{N} \theta_j^2 \tag{23}$$

5.2. Computational process and discrete partitioning

The connection between network nodes is strengthened by the incorporation of the homomorphic encryption method. These techniques are developed on top of the framework of technologies for the Internet of Things (IoT). This tactical use of homomorphic encryption works as an additional layer of defense, ensuring that the secrecy of model changes as well as their integrity are maintained. Before sending their separate contributions to the central server for the purpose of aggregation, each individual node in this procedure encrypts their contributions using a key that is only known to it. The central server is given the ability to perform calculations on encrypted data without first having to decode the data by skillfully using safe addition and multiplication protocols. This enables the server to avoid the need to decrypt the data. This indicates that the data are kept secret even as the computing methods are being carried out. After the aggregation process is complete, the updated versions of the encrypted models are sent back to the nodes from which they originated. When it comes time to carry out the decryption procedure, the nodes use their own private keys that are unique to them. This sequential technique makes it easier to integrate the incoming changes into the appropriate local models in a fluid manner. This method's IoT-driven nature not only ensures a thorough data security framework throughout the whole of the data transmission path, but it also allows collaborative development of models across the linked node network, which is a significant benefit in and of itself. In essence, the integration of IoT-enabled methodologies, combined with the strategic incorporation of homomorphic encryption and localized decryption procedures, establishes a robust and secure foundation. This foundation is capable of supporting collaborative model refinement across a network of interconnected nodes, while ensuring end-to-end data security is maintained at all stages of communication and computation. Individual devices within the network are given in Eq. (24), here i denotes an individual edge device and n denotes the total number of devices within the network. The process of aggregation holds significant importance in the context of federated learning. The process entails the collection or integration of the discrete weights of each model derived from all the devices or nodes present in the network. Each individual device, such as a smartphone, IoT device, or any other edge computing unit, independently trains a local model using its own private data. Following a training session, the local models broadcast their revised weights to a centralized server. The weights are subsequently aggregated by the server, employing methods such as federated averaging, in order to generate a global model. The aforementioned global model effectively assimilates insights derived from various individual devices, thereby establishing a comprehensive repository of collective knowledge. This iterative procedure, which involves training, transmitting updates, and consolidating weights, facilitates the collaborative enhancement of the global model's performance while simultaneously safeguarding the confidentiality of local data.

$$D_i = \{D_1, D_2, \ldots, D_n\} \tag{24}$$

Mathematical methods like secret sharing described in Eqs. (25)–(27) and polynomial interpolation in Eq. (28) are used by the secured addition and multiplication protocols used in multi-party computing to permit secure computation while protecting the privacy of each party's privately shared inputs.

$$E_i = Encrypt(\theta_i, Pk_i) \tag{25}$$

$$\nabla E_i = \frac{1}{n}\sum_{i=1}^{n} E_i \tag{26}$$

$$D_i = Decrypt(\nabla E_i, Sk_i) \tag{27}$$

$$P(x) = \sum_{i=0}^{t} y_i . \prod_{j=0, j=i}^{t} \frac{x - x_j}{x_i - x_j} \tag{28}$$

In the context of our model aggregation process, the notation utilized carries specific meanings. The term E_i denotes the encrypted version of the model update contributed by the node indexed as i. This update is securely encoded to protect its confidentiality during transmission. Correspondingly, θ_i signifies the model update generated by the same node i, encapsulating the refined knowledge derived from local computations. The symbol Pk_i designates the public key unique to node i, which plays a pivotal role in the encryption and decryption procedures. Conversely, Sk_i signifies the private key associated with the same node, instrumental in securely unlocking encrypted data. The aggregate of encrypted updates, denoted as ∇E_i, is the combined encrypted form of all contributions from the various nodes. The parameter n represents the total count of participating nodes in the network. Moving to the realm of multi-party computing, the expression $P(x)$ embodies an interpolated polynomial used for reconstructing shared inputs. This polynomial facilitates the combination of individual contributions while preserving privacy. Within this framework, t denotes the minimum number of parties required to perform secure computations, a key parameter governing the level of privacy preservation. Furthermore, y_i signifies the confidential input contributed by party i, safeguarded through secret sharing mechanisms. Lastly, x_i and x_j are distinct input values associated with different parties, enabling secure interaction within the multi-party protocol. Together, these notations underpin a comprehensive framework for secure collaboration and computation across interconnected nodes while upholding data privacy and integrity. Federated learning involves the training of models on decentralized devices, wherein each device possesses its own local dataset. In order to safeguard privacy, it is imperative to employ encryption techniques to secure sensitive data prior to its transmission. Nevertheless, conventional encryption techniques present a notable obstacle for federated learning as they do not permit calculations on encrypted data without prior decryption. In contrast, Paillier's cryptosystem possesses the property of

homomorphism, hence facilitating the execution of certain operations on encrypted data without necessitating decryption. More precisely, it provides functionality for performing addition and multiplication operations on encrypted values. This characteristic is of utmost importance in activities such as aggregation in federated learning, wherein the central server is required to merge model updates originating from numerous devices while abstaining from seeing the unprocessed data. The process of training and validation of the entire process is represented in Algorithm 1.

The process of federated learning begins with the activation of the central server as the hub of activity. It prepares a global model that will be chosen as M0 and will represent a pre-configured convolutional neural network (CNN) model in its initial state. This model will be used later. This condition is broken out in excruciating detail in the project's Section 4.2.3. Moving on to the topic of safety, the encryption infrastructure is built by first producing a public key (*Pk*) and a secret key (*Sk*) using the dependable Paillier Cryptosystem. This is done before the infrastructure can be used. This cryptographic pair protects the confidentiality of the data during the whole operation. The following phase, which comes after the foundations have been laid, is to ensure that the model's initial parameters are protected. Paillier encryption is applied to each parameter $\theta_i(t)$ connected to the M_0 model. This produces encrypted representations of the parameters that are referred to as E_i. As the procedure moves forward, the attention is redirected to the selection and preparation of nodes, which are discrete units that are responsible for computing. A group of edge devices denoted by D_i is selected at random for each of the 'n' nodes in the network. The data that is stored in these devices goes through the procedure of preprocessing, which ensures that it is ready to be analyzed in greater detail. It is of the utmost importance that the confidential data contained inside these nodes be protected. The Paillier encryption algorithm is used to encrypt the chosen data, which is denoted by X_k. The proper public key, denoted by Pk_i, is used in order to decrypt the data for each edge device. The data's protection is further increased by the sharing of the secret. The $X_{i,k}$ values that have been encrypted are next subjected to secret sharing, which divides the encrypted data into shared secrets that are then spread across the edge nodes.

As the process moves on to the model training phase, the data from several epochs is analyzed by each node so that its own local model may be improved. The grinding process is repeated for each epoch using the batch size (bz) that was previously selected. In order to calculate the local gradients denoted by $\nabla L_i(\theta_G)$, the current model parameters denoted by $\theta_i(t)$, the preprocessed data denoted by X_k, and the shared secrets denoted by D_i are used. These gradients serve as a guide for the updating of the local model

Algorithm 1
Collaborative Learning via IoT-Integrated Federated Multiparty Computation.

1. **Server Initialization:**
   Initialize global model M_0 // M_0: Pre-initialized CNN model at its initial state provided in Section 4.2.3
2. **Public Key Generation:**
   Generate public key *Pk* and secret key *Sk* using Paillier Cryptosystem.
3. **Encryption of Initial Model Parameters:**
   for all parameters $\theta_i(t)$ in M_0:
   E_i= Paillier_Encryption ($\theta_i(t)$, pk)
4. **Node Selection and Data Preparation:**
   for $i = 1$ to n: // Loop over nodes
   Select D_i // Randomly selected edge devices.
   for all $k \in D_i$:
   X_k= Preprocess_Data (X_k)//Preprocessed Data for the k-th edge node
   Encryption of Private Data:
   for all $X_{i,k} \in X_k$:
   Encrypted $X_{i,k}$ = Paillier_Encryption ($X_{i,k}$, Pk_i)
   Encrypted Private Data secret sharing:
   for all $X_{i,k} \in X_k$:
   D_i = Secret_Sharing (Encrypted $X_{i,k}$, over edge nodes)
   Model Training:
   for $q = 1$ to *total_epochs*: // Loop over epochs
   for the designated batch size (*bz*):
   for all $k \in D_i$:
   Compute Local Gradients:
   $\nabla L_i(\theta_G(t))$= Compute_Gradient ($\theta_i(t)$, X_k, D_i, *bz*)
   Update Local Model Weights:
   $\theta_{i(new)} = \theta_G - \mu \nabla L_i(\theta_G)$
   Encryption of Local Model initial weights:
   for all $\theta_{i(new)}$ in H_k: //
   E_i = Paillier_Encryption ($\theta_{i(new)}$, Pk_i)
   Secure Aggregation:
   $\nabla E_i = \frac{1}{n} \sum_{i=1}^{n} E_i$
5. **Aggregated Model Weights Decryption:**
   for all E_i in ∇E_i:
   $E_{i(decrypted)}$= Paillier_Decryption (E_i, Sk_i)
6. **Global Model Updates**
   Update Global Model $\theta_{G(Updated)}$ using $E_{i(decrypted)}$
7. **Return the Updated Global Model to the server:**
   Return $\theta_{G(Updated)}$ to the Server/Cloud

weights $\theta_{i(new)}$, which is accomplished via the use of a learning rate, which promotes convergence. In order to ensure that the revised model weights remain as secure as possible, they have been encrypted once again. The estimated values of $\theta_{i(new)}$ are encrypted using Paillier encryption in a manner that is aligned with the public key Pk_i that has been selected. Secure aggregation emerges as an essential component as a result of the cumulative efforts done across all nodes. The aggregated encrypted weights, denoted by the notation ∇E_i, are the result of aggregating the encrypted local model weights, which entails taking the encrypted values and averaging them across all of the nodes. Discovering how to decipher the aggregated model weights is the pinnacle of this effort. Paillier decryption is performed on each encrypted weight E_i by employing the matching secret key Sk_i. This essentially reveals the modified model parameters in their original, unencrypted form. After the aggregated weights were successfully decrypted, the global model $\theta_{G(Updated)}$ went in for some much-needed revisions. The revised parameters of the model have been included into the overall model, which is now prepared to take into account the findings that have been produced by the distributed computations. In the end, the final stage of this extensive procedure is the process of delivering the updated global model $\theta_{G(Updated)}$. The model incorporates the collective information acquired from all nodes and serves as a valuable resource for informing ongoing decision-making for the devices connected to the network.

6. Result

A visual representation of the model's learning progress may be seen in Fig. 9 showing accuracy and validation accuracy, loss and validation loss, and accuracy and loss. It is crucial to carefully evaluate these numbers in order to comprehend how the model's performance changed over training. Look for patterns that can give you information about how effective the federated learning strategy is, such as convergence, overfitting, or underfitting. An outstanding accuracy of 99.27 % was attained by the global model given in Fig. 9, which reflects the aggregate intelligence of all nodes following the federated learning process. This high level of accuracy indicates that the global model has successfully learnt from the local updates that have been supplied by each edge device. It is a reflection of the quality of cooperation as well as the accumulation of information from the many nodes. The performance of our model is shown in an informative and comprehensive manner in Fig. 9. The first row provides an overview by illustrating how well our global model performs in general, with a particular emphasis on accuracy and loss measures. It serves as an overview of our federated learning journey and demonstrates how the model evolves over time. The truth of the matter is revealed in the rows that follow. Each entry in Fig. 9 starting from row 2 represents a different edge device that is part of the federated network. With this analysis, it is much more efficient to comprehend the unique contributions made by each edge device. The performance of a classification model in accurately discriminating between images labeled as "Fire" and "Non-Fire" can be assessed through the use of a confusion matrix in Fig. 8. The model exhibited a notable level of precision by accurately identifying 109 occurrences of "Non-Fire" images, hence showcasing its resilience in distinguishing safety. Furthermore, the model demonstrated a notable level of competence in accurately identifying images related to "Fire," obtaining a total of 128 instances of correctly identified positive cases. However, the model exhibited two cases in which it did not successfully identify genuine "Non-Fire" images, indicating a minor aspect that may be enhanced. Significantly, the model exhibited a lack of false alarms, indicating that it did not erroneously classify any photos depicting "Non-Fire" as "Fire." In general, the efficiency of the model in reliably categorizing the essential image categories is underscored by the presented confusion matrix.

We are able to assess how well the federated learning approach capitalizes on the distinctive qualities offered by each edge node in the network by comparing the performance of the global model to that of individual devices. This comparison allows us to determine the degree to which the proposed federated learning method is effective. This emphasizes the collaborative nature of our approach and demonstrates the vital role that each edge device plays in the larger design. A graphical example of the synergy that can be seen across

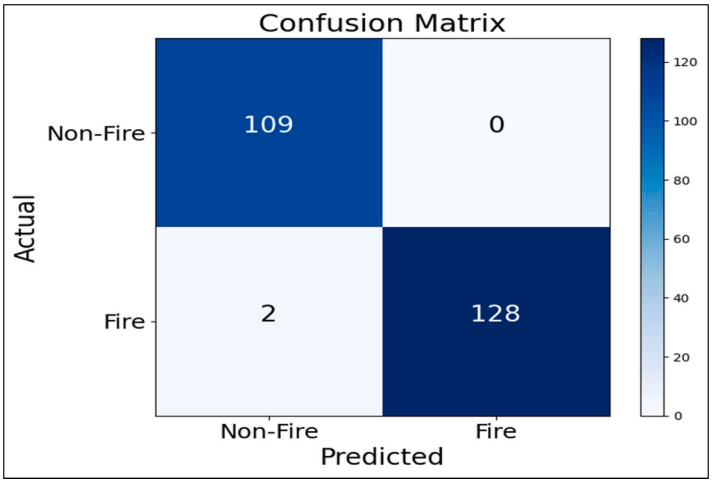


Fig. 8. Confusion Matrix for the proposed model.

our federated learning environment can be found in Fig. 9. It provides a full understanding of how the network as a whole achieves such great accuracy by demonstrating the performance of our model throughout its whole, from the perspective of the entire network to that of individual nodes. Fig. 10 represents the training and validation of Node 4 and 5 respectively.

In Table 5, a full comparison of several approaches and models created for fire detection is provided. Fire detection is an important job that has a wide range of applications, including wildfire monitoring and industrial safety. Each row of the table represents a unique strategy or model. These strategies and models range from well-known architectures like Inceptionv3 and VGG19 to more specialized models like Fire-Net and CNN_S. These approaches are examined in great detail across a variety of important performance measures, each of which is given as a percentage. The fundamental component of model assessment, accuracy evaluates the extent to which predictions are right as a whole. The capacity of the model to reliably detect positive cases is one of the most important aspects that are measured by precision. The F1-Score carefully strikes a compromise between accuracy and recall, while the Recall metric evaluates how well the model is able to accurately identify each and every true positive instance. Notably, the proposed IoT-Enabled FMPC model emerges as the outstanding performer, getting remarkable scores in all measures. This model represents a significant step forward in the industry of fire detection since it has an incredible accuracy rate of 99.27 %, in addition to excellent precision, recall, and F1-Score.

This strategy makes use of the collective intelligence offered by a network of linked nodes, all the while ensuring that the user's personal information and data remain private and secure. Its astounding accuracy of 99.27 %, which demonstrates that it is capable of making extraordinarily exact predictions, is the model's crowning achievement, which makes it a momentous accomplishment. This degree of precision is a game-changer in fields where the repercussions of ignored or false alarms are serious, such as fire detection,

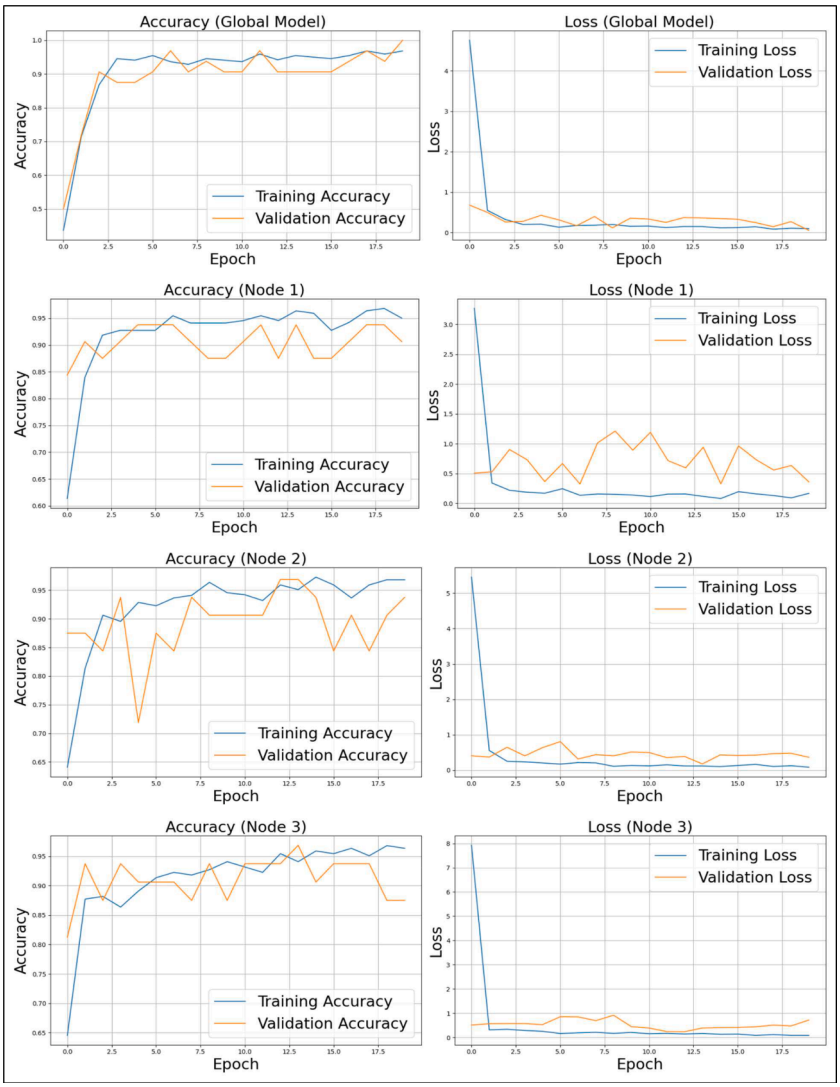


Fig. 9. (Row 1): Global Model Accuracy and Loss. (Row 2): Node 1 Accuracy and Loss. (Row 3): Node 2 Accuracy and Loss. (Row 4): Node 3 Accuracy and Loss.

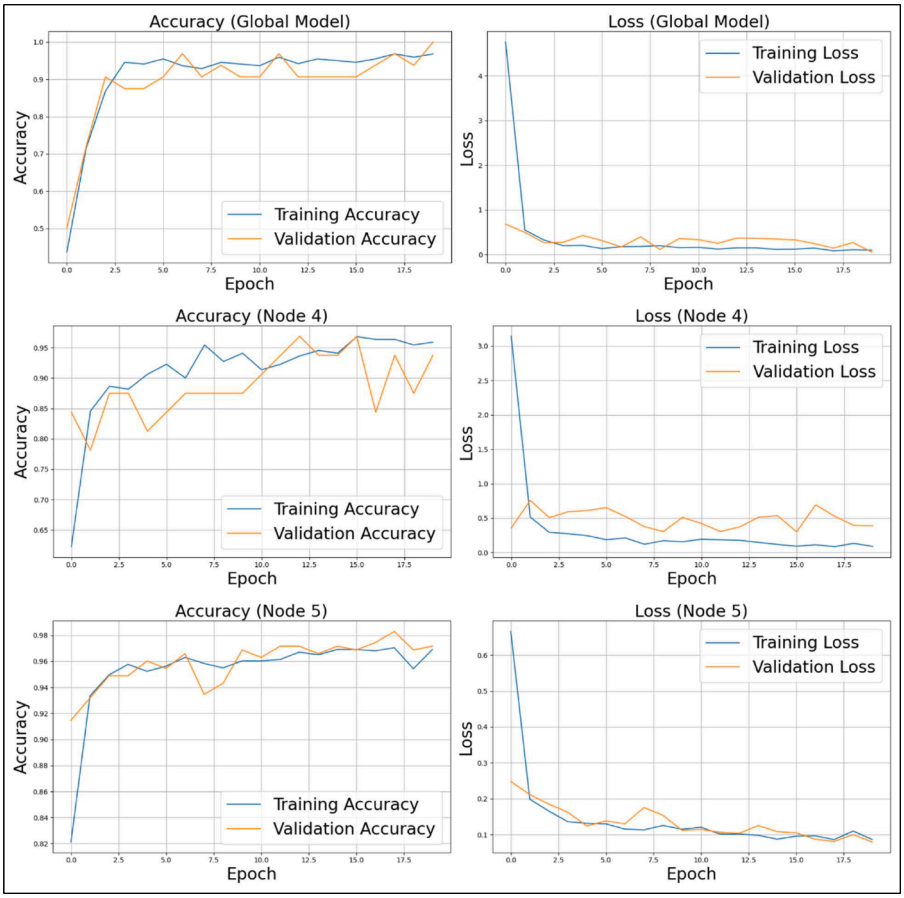


Fig. 10. (Row 1): Global Model Accuracy and Loss. (Row 2): Node 4 Accuracy and Loss. (Row 3): Node 5 Accuracy and Loss.

Table 5
Proposed IoT-enabled strategy compared with the other proposed models.

Authors	Methods	Class dataset	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)
Zhao et al. [33]	Fire-Net	Fire and no-fire	98	–	98.8	–
Sousa et al. [35]	Inceptionv3	Fire and no-fire	93.6	94.1	93.1	–
Sun et al. [39]	CNN_S	Fire and without fire	94.1	–	–	–
Khan et al. [40]	VGG19	Fire and no-fire	95.0	95.7	94.2	94.2
Lee et al. [43]	AlexNet	Fire and no-fire	94.8	–	–	–
	VGG13		86.2			
	Modified VGG13		96.2			
	GoogleNet		99			
	Modified GoogleNet		96.9			
IoT-enabled FMPC **	Federated Learning	Fire and non-fire	99.27	98.92	98.67	99.03

** **Proposed**.

since it reduces the likelihood of either occurring. In addition, the fact that it has an accuracy rate of 98.92 % says eloquently about its capability to properly detect instances of fire, hence assuring that there will be minimum false positives. In addition to this, a recall score of 98.67 % reveals that the model is capable of catching the majority of genuine fire incidences, which helps to reduce the number of false negatives. The model achieves an outstanding 99.03 % on the holistic statistic known as the F1-Score, which reflects the model's harmonious combination of accuracy and recall. In short, the IoT-Enabled FMPC model not only establishes a new benchmark for precision, but it also succeeds in striking the precise balance between precision and recall, which positions it as a durable and trustworthy solution for fire detection in real-world situations with high stakes. Federated Learning involves the execution of computations on edge devices, which often possess constrained resources that may provide challenges in handling extensive pre-trained classifiers such as DenseNet or ResNet. Consequently, the proposed approach utilizes compact classifiers based on CNNs in order to accommodate the limitations imposed by these edge devices. The actual and predicted response generated by the model is given in Fig. 11.

Fig. 11. Few samples of the predicted and actual outputs.

7. Conclusion

IoT-Enabled FMPC is the name of the algorithm that is presented in the interest of expanding the methods that are used to detect fires. It is a model of both innovation and excellence. Detecting forest fires is a mission-critical activity that has a wide variety of real-world applications, ranging from protecting natural ecosystems from the threat of wildfires to assuring the safety of industrial facilities. The proposed algorithm has broken through the conventional barriers and reached high accuracy, precision, recall, and the harmonizing F1-score. It performs very well in accurately identifying "fire" occurrences from "no-fire" cases, hence significantly reducing the number of false alarms. This accuracy, when combined with a high precision rate of 98.92 % and a strong recall of 98.67 %, underlines the usefulness of the system in recognizing genuine fire occurrences while minimizing the number of false warnings. The algorithm's F1-Score of 99.03 %, which is harmonic, illustrates its balanced approach and makes it an important tool in a variety of real-world applications, including the control of wildfires and industrial safety. In addition to its remarkable numerical performance, the proposed IoT-Enabled FMPC algorithm represents the meeting point of cutting-edge technology and fire safety by providing a game-changing solution with the potential to protect people, ecosystems, and essential infrastructure from harm.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgement

The authors extend their appreciation to the Deanship of Scientific Research at King Khalid University for funding this work through Small group Research Project under grant number RGP1/405/44. The authors would like to thank Prince Sultan University for their support.

References

[1] P. De Frenne, J. Lenoir, M. Luoto, B.R. Scheffers, F. Zellweger, J. Aalto, M.B. Ashcroft, et al., Forest microclimates and climate change: importance, drivers and future research agenda, Glob. Chang Biol. 27 (11) (2021) 2279–2297.
[2] V. Arroyo-Rodríguez, L. Fahrig, M. Tabarelli, J.I. Watling, L. Tischendorf, M. Benchimol, E. Cazetta, et al., Designing optimal human-modified landscapes for forest biodiversity conservation, Ecol. Lett. 23 (9) (2020) 1404–1420.
[3] D. Fowler, P. Brimblecombe, J. Burrows, M.R. Heal, P. Grennfelt, D.S. Stevenson, A. Jowett, et al., A chronology of global air quality, Philos. Trans. R. Soc. A 378 (2183) (2020), 20190314.
[4] D.A. Jaffe, S.M. O'Neill, N.K. Larkin, A.L. Holder, D.L. Peterson, J.E. Halofsky, A.G. Rappold, Wildfire and prescribed burning impacts on air quality in the United States, J. Air Waste Manage. Assoc. 70 (6) (2020) 583–615.
[5] S.L. Stephens, A.L.R. Westerling, M.D. Hurteau, M.Z. Peery, C.A. Schultz, S. Thompson, Fire and climate change: conserving seasonally dry forests is still possible, Front. Ecol. Environ. 18 (6) (2020) 354–360.
[6] S. Etzold, M. Ferretti, G.J. Reinds, S. Solberg, A. Gessler, P. Waldner, M. Schaub, et al., Nitrogen deposition is the most important environmental driver of growth of pure, even-aged and managed European forests, For. Ecol. Manage. 458 (2020), 117762.
[7] G. Vieilledent, M. Nourtier, C. Grinand, M. Pedrono, A. Clausen, T. Rabetrano, J.-R. Rakotoarijaona, et al., It's not just poverty: unregulated global market and bad governance explain unceasing deforestation in Western Madagascar, BioRxiv (2020) 2020. -07.
[8] D. Santos, A. Mota, C.F.A. da Silva, S.N. de Melo, P.M. de Almeida Junior, L.F. Bueno, Influence of deforestation inside and outside indigenous lands in the Brazilian Amazon Biome, Reg. Environ. Change 22 (2) (2022) 77.
[9] G. Mataveli, G. de Oliveira, C.H.L. Silva-Junior, S.C. Stark, N. Carvalho, L.O. Anderson, L.V. Gatti, L.E. Aragão, Record-breaking fires in the Brazilian Amazon associated with uncontrolled deforestation, Nature Ecol. Evol. (2022) 1–2.
[10] X. Feng, C. Merow, Z. Liu, D.S. Park, P.R. Roehrdanz, B. Maitner, E.A. Newman, et al., How deregulation, drought and increasing fire impact Amazonian biodiversity, Nature 597 (7877) (2021) 516–521.
[11] P.G. Ladd, X. Zhao, N.J. Enright, Fire regime and climate determine spatial variation in level of Serotiny and population structure in a fire-killed conifer, Plant Ecol. 223 (7) (2022) 849–862.
[12] S. Khan, A. Khan, Ffirenet: deep learning based forest fire classification and detection in smart cities, Symmetry (Basel) 14 (10) (2022) 2155.
[13] A. Kolanek, M. Szymanowski, A. Raczyk, Human activity affects forest fires: the impact of anthropogenic factors on the density of forest fires in Poland, Forests 12 (6) (2021) 728.
[14] J. Paudel, Short-run environmental effects of COVID-19: evidence from forest fires, World Dev. 137 (2021), 105120.
[15] S. Sannigrahi, F. Pilla, A. Maiti, S. Bar, S. Bhatt, Q. Zhang, S. Keesstra, A. Cerda, Examining the status of forest fire emission in 2020 and its connection to COVID-19 incidents in West Coast regions of the United States, Environ. Res. 210 (2022), 112818.
[16] A. Gupta, C.M. Bhatt, A. Roy, P. Chauhan, COVID-19 lockdown a window of opportunity to understand the role of human activity on forest fire incidences in the Western Himalaya, India, Curr. Sci. 119 (2) (2020) 390–398.
[17] J.A. Stanturf, N. Mansuy, COVID-19 and forests in Canada and the United States: initial assessment and beyond, Front. Forest. Glob. Change 4 (2021), 666960.
[18] R. Chen, Y. Luo, Mohd.R. Alsharif, Forest fire detection algorithm based on digital image, J. Softw. 8 (8) (2013) 1897–1905.
[19] K. Angayarkkani, N. Radhakrishnan, Efficient forest fire detection system: a spatial data mining and image processing based approach, Int. J. Comput. Sci. Netw. Secur. 9 (3) (2009) 100–107.
[20] E.A. Kadir, H. Irie, S.L. Rosa, Modeling of wireless sensor networks for detection land and forest fire hotspot, in: 2019 International Conference on Electronics, Information, and Communication (ICEIC), IEEE, 2019, pp. 1–5.
[21] U. Dampage, L. Bandaranayake, R. Wanasinghe, K. Kottahachchi, B. Jayasanka, Forest fire detection system using wireless sensor networks and machine learning, Sci. Rep. 12 (1) (2022) 46.
[22] K. Grover, D. Kahali, S. Verma, B. Subramanian, WSN-based system for forest fire detection and mitigation, in: Emerging Technologies for Agriculture and Environment: Select Proceedings of ITsFEW 2018, Springer Singapore, Singapore, 2019, pp. 249–260.
[23] A. Dharmawan, A. Harjoko, F.D. Adhinata, Region-based annotation data of fire images for intelligent surveillance system, Data Brief 41 (2022), 107925.
[24] A.A. Siddique, S.M. Umar Talha, M. Aamir, A.D. Algarni, N.F. Soliman, W. El-Shafai, COVID-19 classification from X-Ray images: an approach to implement federated learning on decentralized dataset, Comput. Mater. Continua (2023) 3883–3901.
[25] Z. Liu, K. Zhang, C. Wang, S. Huang, Research on the identification method for the forest fire based on deep learning, Optik (Stuttg) 223 (2020), 165491.
[26] A.A. Siddique, W. Boulila, M.S. Alshehri, F. Ahmed, T.R. Gadekallu, N. Victor, M. Tahir Qadri, J. Ahmad, Privacy-enhanced pneumonia diagnosis: IoT-enabled federated multi-party computation in industry 5.0, IEEE Trans. Consum. Electron. (2023).
[27] J.-L. Shin, W.-W. Seo, T. Kim, J. Park, C.-S. Woo, Using UAV multispectral images for classification of forest burn severity—a case study of the 2019 Gangneung forest fire, Forests 10 (11) (2019) 1025.
[28] Q.-X. Zhang, G.-H. Lin, Y.-M. Zhang, G. Xu, J.-J. Wang, Wildland forest fire smoke detection based on faster R-CNN using synthetic smoke images, Procedia Eng 211 (2018) 441–446.
[29] F.M.A. Hossain, Y.M. Zhang, M.A. Tonima, Forest fire flame and smoke detection from UAV-captured images using fire-specific color features and multi-color space local binary pattern, J. Unmanned Veh. Syst. 8 (4) (2020) 285–309.
[30] K. Srinivas, M. Dua, Fog computing and deep CNN based efficient approach to early forest fire detection with unmanned aerial vehicles. Inventive Computation Technologies 4, Springer International Publishing, 2020, pp. 646–652.
[31] Z. Jiao, Y. Zhang, L. Mu, J. Xin, S. Jiao, H. Liu, D. Liu, A yolov3-based learning strategy for real-time uav-based forest fire detection, in: 2020 Chinese Control and Decision Conference (CCDC), IEEE, 2020, pp. 4963–4967.
[32] R. Kaabi, M. Sayadi, M. Bouchouicha, F. Fnaiech, E. Moreau, J.M. Ginoux, Early smoke detection of forest wildfire video using deep belief network, in: 2018 4th International Conference on Advanced Technologies for Signal and Image Processing (ATSIP), IEEE, 2018, pp. 1–6.
[33] Y. Zhao, J. Ma, X. Li, J. Zhang, Saliency detection and deep learning-based wildfire identification in UAV imagery, Sensors 18 (3) (2018) 712.
[34] Y. Cao, F. Yang, Q. Tang, X. Lu, An attention-enhanced bidirectional LSTM for early forest fire smoke recognition, IEEE Access 7 (2019) 154732–154742.
[35] M.J. Sousa, A. Moutinho, M. Almeida, Wildfire detection using transfer learning on augmented datasets, Expert Syst. Appl. 142 (2020), 112975.
[36] Y. Tang, H. Feng, J. Chen, Y. Chen, ForestResNet: a deep learning algorithm for forest image classification, J. Phys.: Conf. Ser. 2024 (1) (2021), 012053.

[37] M. Park, D.Q. Tran, S. Lee, S. Park, Multilabel image classification with deep transfer learning for decision support on wildfire response, Remote Sens. (Basel) 13 (19) (2021) 3985.
[38] K. Govil, M.L. Welch, J. Timothy Ball, C.R. Pennypacker, Preliminary results from a wildfire detection system using deep learning on remote camera images, Remote Sens. (Basel) 12 (1) (2020) 166.
[39] X. Sun, L. Sun, Y. Huang, Forest fire smoke recognition based on convolutional neural network, J. Forest. Res. 32 (5) (2021) 1921–1927.
[40] A. Khan, B. Hassan, S. Khan, R. Ahmed, A. Abuassba, DeepFire: a novel dataset and deep transfer learning benchmark for forest fire detection, Mobile Inf. Syst. 2022 (2022).
[41] F. Ma, Y. Li, S. Ni, S.L. Huang, L. Zhang, Data augmentation for audio-visual emotion recognition with an efficient multimodal conditional GAN, Appl. Sci. 12 (1) (2022) 527, https://doi.org/10.3390/app12010527.
[42] K. Ding, Z. Xu, H. Tong, H. Liu, Data augmentation for deep graph learning: a survey, ACM SIGKDD Expl. Newsletter 24 (2) (2022) 61–77.
[43] W. Lee, S. Kim, Y.-T. Lee, H.-W. Lee, M. Choi, Deep neural networks for wild fire detection with unmanned aerial vehicle, in: 2017 IEEE International Conference on Consumer Electronics (ICCE), IEEE, 2017, pp. 252–253.