mathematics

MDPI

Article

Distributed Fire Detection and Localization Model Using Federated Learning

Yue Hu [1,†], Xinghao Fu [1,†] and Wei Zeng [2,*]

[1] School of Automation Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China
[2] School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China
* Correspondence: zwei504@uestc.edu.cn
† These authors contributed equally to this work.

Abstract: Fire detection and monitoring systems based on machine vision have been gradually developed in recent years. Traditional centralized deep learning model training methods transfer large amounts of video image data to the cloud, making image data privacy and confidentiality difficult. In order to protect the data privacy in the fire detection system with heterogeneous data and to enhance its efficiency, this paper proposes an improved federated learning algorithm incorporating computer vision: FedVIS, which uses a federated dropout and gradient selection algorithm to reduce communication overhead, and uses a transformer to replace a traditional neural network to improve the robustness of federated learning in the context of heterogeneous data. FedVIS can reduce the communication overhead in addition to reducing the catastrophic forgetting of previous devices, improving convergence, and producing superior global models. In this paper's experimental results, FedVIS outperforms the common federated learning methods FedSGD, FedAVG, FedAWS, and CMFL, and improves the detection effect by reducing communication costs. As the amount of clients increases, the accuracy of other algorithmic models decreases by 2–5%, and the number of communication rounds required increases significantly; meanwhile, our method maintains a superior detection performance while requiring roughly the same number of communication rounds.

Keywords: federated learning; fire detection system; privacy preservation; deep learning

MSC: 68T07; 68T45

Citation: Hu, Y.; Fu, X.; Zeng, W. Distributed Fire Detection and Localization Model Using Federated Learning. *Mathematics* **2023**, *11*, 1647. https://doi.org/10.3390/math11071647

Academic Editor: Alessandro Niccolai

Received: 14 February 2023
Revised: 15 March 2023
Accepted: 27 March 2023
Published: 29 March 2023

1. Introduction

Fire is one of the disasters that most frequently and commonly threatens public safety and social development. According to the Fire and Rescue Bureau of China's Ministry of Emergency Management's 3 April 2022 briefing, 219,000 fires were reported in China [1] in the first quarter of this year, resulting in 625 deaths, 397 injuries, and 1.52 billion CNY in direct property damage due to fire. As a result, early fire warning is critical. Sensors are used in traditional fire detection methods to detect smoke, fire scale, the initial flame area, atmospheric temperature, and so on. Because of their low cost and ease of use, these sensors have been widely utilized. However, there are numerous issues and pitfalls with this traditional detection method. For example, smoke takes time to reach the ceiling, resulting in late alarm triggering and failure to warn of a fire in a timely manner. Furthermore, many sensors perform best in confined spaces and are difficult to use in open areas. In recent years, machine vision-based fire detection and monitoring systems have gradually developed, with lower costs, better real-time performance, and the ability to achieve high detection accuracy.

However, these fire detection methods still face some difficulties. The privacy of images captured by smart cameras is compromised during upload to the cloud. Furthermore,

the traditional centralized deep learning model training method is heavily reliant on data collection and fusion. If we are unable to obtain complete and rich data to train the model and develop technology, the effect of the deep learning model will be severely constrained. Nowadays, the contradiction between the phenomenon of data islands and the demand for data fusion is becoming increasingly apparent. Federated learning methods can avoid directly exposing data to third parties by interacting only with local model parameters, which has a natural protective effect on data privacy and prevents data leakage.

Federated learning is one approach to the issue of data security and privacy. Federated learning trains centralized models using decentralized data. Traditional centralized learning methods cannot protect user data and are difficult to implement in real-time. Federated learning's distributed model training, on the other hand, keeps user data in the local area and performs model training and deployment locally on the device. The server side iterates the global model update by aggregating the encrypted client model parameters. The new model parameters are then sent down to the local level, where individual clients work together to update a better, more comprehensive model, allowing the model to evolve from the local optimum to the global optimum.

Despite the growing interest in federated learning research, few works have been published. Additionally, different cameras, different client environments, and other factors contribute to the problem of data heterogeneity. In this paper, an improved federated learning algorithm (FedVIS) is used to collaboratively train distributed models on each distributed client, with model parameters fused interactively in the cloud. In this paper, we propose a federated learning and machine vision-based fire detection and localization system.

The approach's main contributions are summarized below:

1. The use of federated learning techniques effectively addresses the data security and privacy issues in fire detection. The traditional centralized deep learning model training method requires the transmission of a large amount of video and image data to the cloud, which not only takes up a large amount of network bandwidth but also makes it difficult to ensure the privacy and confidentiality of image data; whereas, the most recent federated learning technique can effectively resolve these issues. Federated learning, as a privacy-preserving, secure encryption technology-based distributed machine learning framework, meets the needs of collaborative machine model learning training without disclosing private client data. Federated learning offers a method for preserving user privacy by decentralizing data from the central server to the client [2].
2. Improved performance of federated learning for handling heterogeneous data. Using the transformer model instead of a traditional neural network in federated learning can accelerate convergence when dealing with heterogeneous data from different fire detection environments and different client cameras. The main area for development is the improved robustness of the transformer model for heterogeneous data, and the self-attention-based transformer is more robust to distribution changes, resulting in a superior global model.
3. The FedVIS algorithm reduces communication overhead in federated learning detection and enables more robust and stable federated learning in heterogeneous federated networks. In this paper, an improved federated learning algorithm FedVIS algorithm is proposed to reduce the number of parameters that must be passed from the server to the client using the federated dropout algorithm, and to further accelerate the training by using the gradient selection algorithm to upload only gradients with a higher relevance to the global gradient at the client side.

2. Related Work

2.1. Fire Detection

Fire is recognized as a great hazard, and the monitoring of fire is necessary. In the research on fire detection, vision-based sensors are more accurate and give fewer false

alarms than traditional sensor-based methods such as light, heat, humidity, and gas sensors. This is due to the fact that vision-based sensors have superior classification precision and faster response times [3,4]. However, in certain environments with particularly stringent data confidentiality requirements, these vision-based sensors are extremely intrusive of privacy and are highly insecure [5].

This paper addresses the issue of privacy protection in the fire detection system. The literature in this field is reviewed along two dimensions: vision-based fire detection system literature and privacy protection literature.

In the study of vision-based fire detection systems, the performance of fire detection systems is frequently improved by concentrating on data-understanding algorithms. Traditional techniques for detecting fires using vision rely on three primary classification steps: (1) fire zone characterization, (2) edge detection, and (3) classification. In 2018, Muhammad et al. [6] studied four different fire detection models using convolutional neural networks for early fire detection. They used pre-trained AlexNet CNN models and fine-tuned them to detect fires in various indoor and outdoor environments. The literature [7] proposed a CNN-based architecture by fine-tuning the GoogleNet, where they tried to balance the efficiency and accuracy of the model. They showed that their model could detect the fire properly, even though video frames could be affected by noise or contain a small fire. In 2019, the literature [8] proposed a CNN-based method for detecting fires in captured video frames in uncertain IoT environments. In order to reduce computational complexity and storage, they employed lightweight deep neural networks without dense, fully connected layers. One of the most popular architectures used to construct deep neural networks in several fields, including fire detection, is convolutional neural network architecture. In order to solve the problems of limited accuracy, and high calculation cost of some models which require a powerful system to run, an excellent and efficient SE-EFFNet model for fire detection is created by Khan et al. [9]. In 2021, Xu et al. [10] developed a novel integrated learning approach to detect forest fires in various settings. Yolov5 and EfficientDet, two independent object detectors, were combined to complete the fire detection process. Furthermore, they employed EfficientNet, an individual learner responsible for acquiring global information to prevent false alarms. The detection results are ultimately determined by the decisions of the three learners. None of the above methods is aimed at the privacy protection of fire detection.

In the study of privacy protection, S. Liu et al. [11] used image editing for encryption and image processing of individuals' faces to protect people's identities. However, the privacy leakage of information other than the face in the photograph remains a concern. In addition to this, with current technology, the original image can be recovered from the altered version. Z. Luo et al. [12] used thermal cameras in elderly care applications to solve the privacy problem by capturing images with severely impaired visibility. However, the cost of thermal cameras is too high, and large-scale deployment of thermal cameras is impractical. Ankit and Abhishek [13] suggested a vision-based fire detection system for monitoring private places that use near-infrared (NIR) cameras to collect photos with low visibility, thus maintaining the privacy of users while surveying local and global audiences by using a crowdsourcing service to assess the acceptable level of privacy in the images. However, another problem with privacy is its subjective nature. In terms of personal privacy protection, images with low visibility captured by near-infrared (NIR) cameras may be perfectly acceptable. In some circumstances, however, where data security and privacy issues are of the utmost importance, even infrared images captured in conditions of severely reduced visibility run the risk of compromising confidentiality.

2.2. Federated Learning

In recent years, federated learning has been a significant subject of study, with numerous applications in the industry. Google first proposed federated learning in April 2017 [14]. Individual participants can co-construct models without disclosing the underlying data or its encrypted form. Federated learning minimizes privacy and communication issues

in machine learning tasks and reduces the need to aggregate all data on a single client, and enables machine learning models to learn decentralized data held on individual users (clients). Many clients with high-quality data select federated learning systems for their unique privacy-protecting measures. With federated learning, data silos can be efficiently broken down to make data more useful and establish a win–win situation with guaranteed privacy for many consumers.

In this paper, we combine federated learning to address data privacy issues in machine vision-based fire detection systems. Federated learning for image processing ensures the security of the data required to train the model. The primary benefits of federated learning in image processing applications [15] are real-time prediction; protecting data privacy and security; supporting offline prediction; and having an intelligent framework. In the case of federated learning, all predictions are performed on the edge device; therefore, there is no cause for concern over data transfer delays. Additionally, since federated learning conducts its own training, only the model needs to be transferred. Federated learning offers a method for preserving user privacy by decentralizing data from the central server to the client. Two key factors led to the development of this paradigm [2]: (1) The lack of adequate data to be stored centrally on the server-side owing to direct access limitations on such data. (2) Using local data from clients to secure sensitive data rather than forwarding it to the server, if network asynchronous communication is involved. Even if the device is offline, the process of prediction continues. Consequently, online or offline, there is no need for concern regarding the device. As long as the input devices are available, the model can use them to complete the task. In addition, the infrastructure requirements are minimal because federated learning does not require any type of complex hardware to operate.

Federated learning enables edge devices to train generic models without sending raw data to the cloud [16]. In a machine vision-based fire detection system, the edge devices are cameras. The recorded dataset from the edge device is sent to the cloud so that a model may be trained on the merged data. After that, the edge device receives the trained model for inference. This strategy carries a considerable danger of jeopardizing the privacy of sensitive user data, such as video recordings. Therefore, the distributed machine-learning approach of federated learning can successfully offset this tendency. It may be viewed as a distributed model training based on samples in horizontal federated learning [2]. All data are distributed among various clients. Each client obtains the model from the server, and various devices. Each machine obtains the model from the server, trains the model using local data, and then sends back the modified parameters to the server. Each client's provided parameters are aggregated by the server, which then updates the model and sends each client the most recent version. Each client in this operation is a fully functional replica of the other, and there is no inter-client communication. Each client has the ability to work independently while doing so. In order to train a general global model, locally optimized model weights are pooled centrally. Because the raw data never leave the client and only a subset of model weights are communicated to the cloud, some privacy problems are handled by design. Federated learning is, as a result, a fundamentally privacy-preserving method.

The following are the common federated learning algorithms.

FedSGD algorithm [17] delivers the training results from the client side to the server side after conducting local training using local data. It participates in the average joint aggregation by waiting. This waiting process may lead to a long wait due to many reasons, such as network delays and equipment failure of some clients. The steps of the FedSGD algorithm are as follows: 1. Select some nodes in each batch, conduct an epoch training, and then upload each node to the server. 2. The server will add and sum all the weight to obtain a new weight and distribute it to each node. 3. Each node will replace the distributed node with the weight calculated by the previous epoch to train the new epoch. Repeat the above three steps until the server determines weight convergence.

The FedAVG algorithm [18] is based on FedSGD to divide the original data into multiple parts on the client side. It is the most prevalent method for model optimization

in federated learning. This method averages the locally uploaded stochastic descent gradient data before updating and redistributing it locally. It has been shown to perform well in multi-task learning. However, the FedAvg algorithm itself has some drawbacks: it suffers from global model instability and sluggish convergence when dealing with heterogeneous data. Additionally, the existing federated learning methods (such as the classical federated learning algorithms FedSGD, FedAVG, etc.) face the problem of low computational efficiency when they are applied to the scene of a fire monitoring system due to the characteristics of long training time and poor cooperative training effect.

Wang et al. [19] proposed an algorithm called Communication Mitigated Federated Learning (CMFL) for the local model updates uploaded by the client which contains a large amount of redundant and irrelevant information, which severely occupies the communication bandwidth; therefore, the algorithm requires the client to filter the local model updates with the previous round of global model correlation, and to avoid uploading local model updates that fail to meet the threshold requirement by the percentage of model gradients with the same value.

In a federated environment where each client has access to only one class's positive data, researchers are thinking about developing a multi-class classification model. As a result, the clients must locally update the classifier throughout each federated learning training process without having access to the data and model parameters for the negative classes. So, utilizing traditional decentralized learning methods carelessly, such as distributed SGD or Federated Averaging, may result in excessively simple or subpar classifiers. All of the class embeddings may come to rest at one moment, especially for embedding-based classifiers. To address this problem, as a universal training framework employing just positive class labels, the Federated Averaging with Spreadout (FedAWS) algorithm [20] was presented. A geometric regularizer is added by the server after each iteration to help with the distribution of classes in the embedding space.

Due to the absence of high-quality labeled data created from real-world edges for federated learning, the majority of present work still simulates data federated applications using public datasets and manual partitioning. Thus, benchmarking and model evaluation of federated learning have lagged behind [21]. Researchers summarized the problems faced by federated learning in practical applications as follows [22]: (1) There is heterogeneity in storage, computing, and communication capabilities of each client (device). (2) The issue of data heterogeneity is brought on by each client's non-independent and identically distributed (Non-IID) local data. (3) The model heterogeneity that each client requires is based on its application circumstance. In a federated learning-based fire detection system, there is a non-identical distribution because of the large differences in data distribution. In addition, due to factors such as user groups and geographical associations, the data distribution of these devices is often related, which is non-independent. Recent research has shown that replacing traditional neural networks in federated learning with transformer can considerably reduce catastrophic forgetting, expedite convergence, and build superior global models, especially when dealing with heterogeneous data [23]. The objective of federated learning is to train machine learning models using private data from a large number of distributed devices. Data heterogeneity in the scenario of fire detection is generated by factors such as the employment of different cameras, monitored settings, and levels of illumination. Parallel federated learning approaches display unguaranteed convergence and model weight scattering due to the spread of training data among numerous clients. In this paper, we employ the transformer instead of a conventional neural network for federated learning to address this issue.

The objective of federated learning is to train machine learning models using private data from a large number of distributed devices. Data heterogeneity in the scenario of fire detection is generated by factors such as the employment of different cameras, monitored settings, and levels of illumination. Parallel federated learning approaches display unguaranteed convergence and model weight scattering as a result of the spread of

training data among numerous clients. Therefore, we employ the transformer instead of a conventional neural network for federated learning to address this issue.

3. Methods

3.1. Transformer in Federated Learning

This research develops a vision-based privacy-preserving and effective fire detection system to address the security issue of image information leakage in fire detection and to detect fires while protecting privacy. A significant portion of client camera data cannot be made public due to privacy concerns; hence, current solutions suffer from the problem of sparse datasets. Due to the limited amount of the dataset, the high-dimensional input space corresponding to the tiny sample size is sparse, making it challenging for neural networks to discover mapping correlations from it. Training the neural network can so easily result in overfitting. In this paper, generated Gaussian noise is utilized to enhance the model's robustness and generalization to diverse input, hence facilitating neural network learning.

In addition, the objective of federated learning is to train machine learning models using private data from a large number of distributed devices. Data heterogeneity in the scenario of fire detection is generated by factors such as the employment of different cameras, monitored settings, and levels of illumination. Parallel federated learning approaches display unguaranteed convergence and model weight scattering as a result of the spread of training data among numerous clients. Therefore, we employ the transformer instead of a conventional neural network for federated learning to address this issue.

The architecture of our method is demonstrated in the Figure 1. Transformer, an attention-based encoder-decoder architecture, has revolutionized not only the field of natural language processing (NLP), but also the field of computer vision. The input to the transformer family of models is a sequence of tokens, denoted by $x_i{}_{(i=1)}^{|x|}$, and this token word embedding is represented by the matrix $X_0 = [x_1, \ldots, x_{|x|}]^T$. There are stacked multiple layers in the transformer, and the transformer computes the output of the final lth layer $X_l = transformer_l(X_{(l-1)}), l \in [1, L]$. The core part of the transformer is the multi-headed attention mechanism, and the computation process of the h th attention head can be expressed as

$$Q_h = XW_h^Q, K_h = XW_h^K, V_h = XW_h^V \tag{1}$$

$$A_h = \frac{(Q_h\, K_h^T)}{\sqrt{(}d_k)} \tag{2}$$

$$H_h = AttentionHead(X) = A_h\, V_h \tag{3}$$

Here, $Q_h, K_h \in R^{n \times d_k}, V_h \in R^{n \times d_k}$ and $A_{i,j}$ denotes the attention weight of $token x_j$ to x_j. They are the Query (Q), the Key (K), the Value (V), and attention (A). Q,K, and V are three different weight matrices by the embedding vector X which is multiplied by 3 different weight matrices W_Q, W_K, W_V. Usually there are multiple heads; the number is denoted as $|h|$, and then the computation result of the multi-head attention mechanism is denoted as $MultiH(X) = [H_1, \ldots, H_{|h|}]W^o, W^o \in R^{|h|d_v \times d_x}$, where [.] represents the splicing operation.

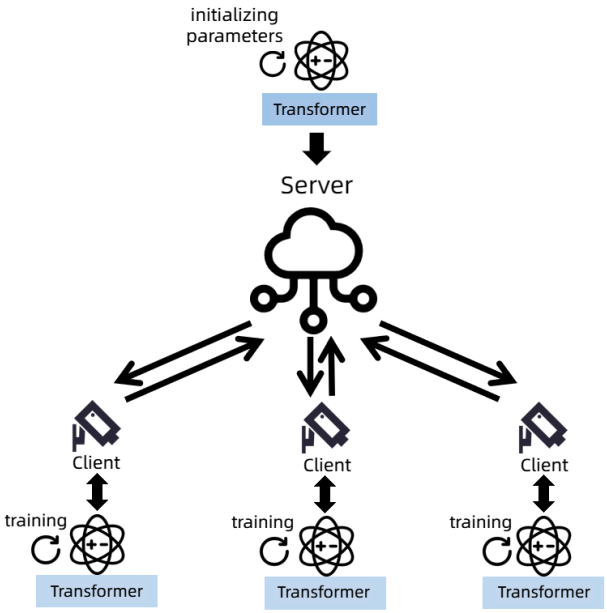


Figure 1. Replacing Traditional Neural Network with Transformer.

In comparison to convolutional neural networks (CNN), visual transformer (ViT) focuses on superior modeling skills to produce an exceptional performance on ImageNet, COCO, and other datasets. The multi-head self-attention (MSA) block, the position encoding block, and the multi-layer perceptron (MLP) block make up ViT's structural components [24]. The following are the formulas for ViT.

$$X \leftarrow MSA(LN(X)) + X \tag{4}$$

$$X \leftarrow MLP(LN(X)) + X \tag{5}$$

Prior to the first MSA, the input is added with the position encoding block. LN is the layer-normalization layer. The following is the formulation of the MSA mechanism.

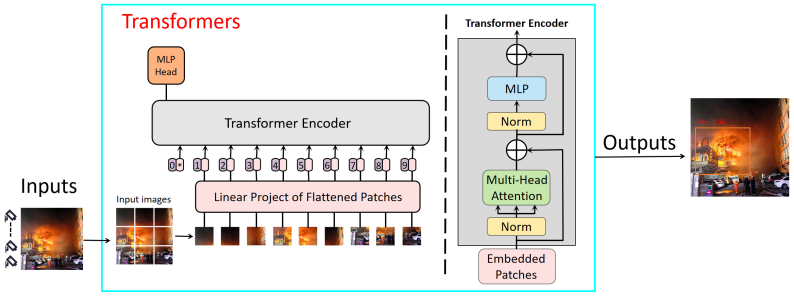
$$MSA(X) = FC_{\text{out}}\left(\text{Attention}\left(FC_q(X), FC_k(X), FC_v(X)\right)\right) \tag{6}$$

$$\text{Attention}\ (Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V \tag{7}$$

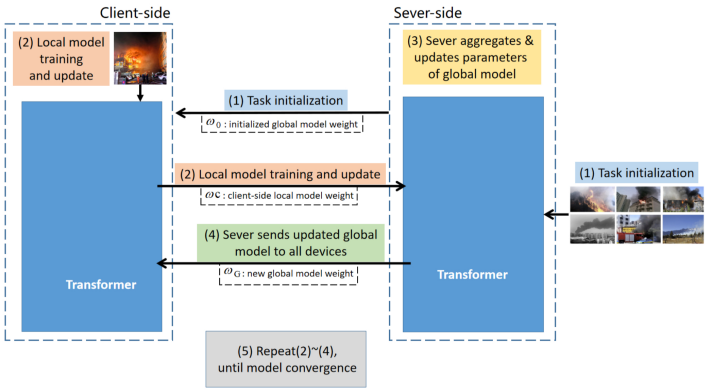
The floating-point operations per second (FLOPs) of MSA and MLP are $4nc^2 + 2n^2c$ and $8nc^2$, respectively, when the hidden dimension of MLP is by default set to $4c$. The ViT with L blocks has $L(12nc^2 + 2n^2c)$ FLOPs.

Recent research has shown that replacing traditional neural networks in federated learning with transformer can considerably reduce catastrophic forgetting, expedite convergence, and build superior global models, especially when dealing with heterogeneous data [23]. The transformer model's enhanced robustness to heterogeneous data, which significantly reduces the catastrophic forgetting of previous devices during training on various new devices, is the main area of improvement. Visual transformer (ViT) is resistant to a variety of corruptions, a characteristic attributable in part to the self-attention process. Self-attention enhances naturally forming clusters in the token, as confirmed by Zhou D et al. [25].

As shown in Figure 2, it is the improved architecture in this paper. In the federated learning architecture, we use the transformer model to replace the traditional neura network.



(a) Vision Transformer on the client and server.



(b) Transformer in federated learning.

Figure 2. Vision Transformer in Federated Learning. Figure (**a**) is the general workflow of Vision Transformer on the client and server. In figure (**b**), the steps of the transformer in federated learning are shown.

Transform is the basic object detection algorithm of the whole system. The main body of a transmission is the parameters of the transformer model. Here are the steps:

1. The server trains first and sends the model parameters to the client;
2. The clients continue training with their local dataset on this basis;
3. The server aggregates and updates parameters;
4. The server distributes the global model to the clients;
5. Repeat, until model convergence.

3.2. Improved Federated Learning Algorithm

Federated learning for edge networks in machine vision-based fire detection systems necessitates interaction with multiple edge nodes. Furthermore, client and server wireless communications are frequently sluggish or unreliable, and bandwidth resources are more limited; so, we wish to reduce the number of client–server communication rounds. The amount of the dataset on each device, however, is less than the total size of the dataset, and the processor speed of the edge device is relatively quick. Consequently, the calculation cost of many model types is typically ignored in comparison to the cost of communication. In this instance, the number of communication rounds is a performance bottleneck for the entire learning framework [26].

In order to ensure that the correctness of the final global model is not compromised, even if communication costs are saved, this research offers an enhanced federated learning approach called FedVIS. The gradient selection method and the federated dropout algorithm are used to lower the communication cost of federated learning. Furthermore, it reduces the downlink communication overhead between the central server and the client. In addition, it may be effortlessly linked with the uplink communication overhead handling mechanism. It reduces the cost of communication without diminishing the accuracy of the final global model. Simultaneously, the model's complexity is lowered, hence increasing

its generalizability. We selected to upload K gradient values based on the literature cited in [27]. The criterion for selection is the Pearson product-moment correlation coefficient, which evaluates the association between the global gradient and the client gradient. Then, only the gradient with the highest correlation is uploaded. This allows the gradient value uploaded by the client to be constrained to be closer to the global gradient value. It reduces not only the cost of calculations but also the cost of communication, based on the concept of maintaining model convergence. In addition, it theoretically guarantees model convergence when data are not independent and distributed equally, and it is more durable and stable in heterogeneous federated networks. In addition, in the Federated Dropout algorithm [28], each client learns smaller submodels, which are subsets of the global model rather than training updates to the complete global model locally. Consequently, the communication burden in federated learning is greatly decreased. Compared to other federated learning algorithms, the FedVIS algorithm has the following advantages:

(1) Saving the server-to-client communication overhead. This study introduces the Federated Dropout technique to significantly reduce communication costs. Instead of training the update of the global model locally, each client simply trains the update of a submodel. In the conventional Dropout method, a random binary mask is multiplied with the hidden units in order to reject a percentage of the desired neurons each time training is relayed through the network. Because the mask differs between processes, each process must calculate the gradient relative to a unique submodel. Based on the number of neurons deleted from each layer, these submodels can have varied sizes (structures). To reduce communication overhead in federated learning, we cancel a predetermined amount of activations on each fully linked layer such that all submodels have the same simple architecture.

As shown in Figure 3, we choose a random activation from each layer to discard, producing a submodel with a 2 × 2 dense matrix.

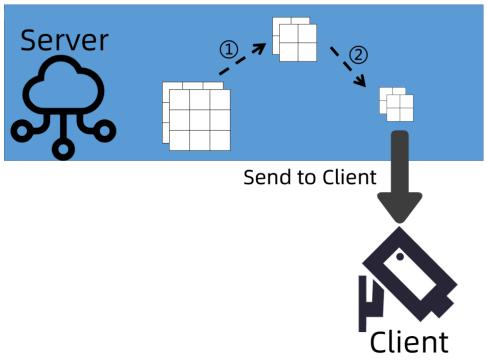


Figure 3. Server to Client. (1) The sub model is constructed through Federated Dropout, and (2) the generated objects are lossless compressed to reduce the size of the communication model.

Therefore, only the necessary coefficients are transferred to the client and repackaged into smaller dense matrices.

(2) The cost of local computation is reduced further in federated learning. Additionally, the size of client-to-central server updates is decreased, and the local training procedure requires fewer gradient updates. As depicted in Figure 4, in addition to saving the communication cost from server to client, Federated Dropout also brings two other benefits. First, the scale of client-to-central server updates has also decreased. Secondly, the local training process now only needs to run fewer gradient updates [28]. At present, all matrix multiplication methods are of smaller dimensions (relative to the full connection layer) or only need to use a few filters (for the convolution layer). Therefore, the use of Federated Dropout further reduces the local computing costs in federated learning.

The client, uninformed of the original model's design, trains its sub-model and provides its updates; then, the central server maps the updates back to the global model. For the convolutional layer, zeroing out the activation does not result in any space savings;

therefore, a portion of the filter is eliminated. Therefore, all current matrix multiplications are of lower dimensionality or employ fewer filters.

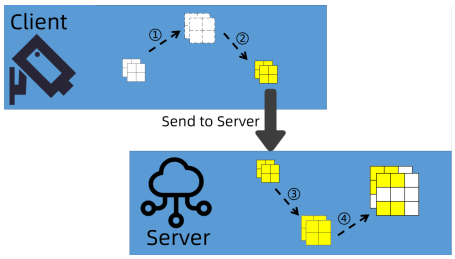


Figure 4. Client to Server. (1) The client decompresses and trains the compression model using local data, and (2) compresses the final local update. They send the local updates back to the central server, and (3) decompression is carried out by the central server, and (4) the global model is created through aggregation on the central server.

(3) Choosing partial gradient uploads for each client will speed up the training. In the Internet context, where the uplink speed is significantly slower than the download connection speed, the interactive communication time is primarily focused on the gradient data upload phase. According to the research [29], it is shown that 99% of ladder interaction communication is redundant in machine learning models with distributed architecture. These duplicate gradients impose a non-negligible communication burden on the parameter server, which can be substantial. In addition, the data between devices is diverse due to differences in fire monitoring camera models, monitoring conditions, and other aspects. If the gap between each client's optimized model and the initial global model supplied by the server is too great, the local model will diverge from the original global model. It will slow the global model's convergence. Based on the FedAvg loss function, LI Tian et al. [28] developed a proximal term to ensure that the model parameters produced by the client after local training are not too different from the initial server values. In the FedVIS algorithm, only gradients having a high correlation with the global gradient are selected for uploading to the server, thereby further accelerating training and decreasing communication time.

Specifically, the global gradient vector w is first solved on the server side, and then the Pearson coefficients of w_0 and $g[j]$ can be derived sequentially after the client side calculates the gradient vector g in each training.

$$\rho_{w_0, g[1]} = \frac{cov(w, g[1])}{\sigma_w \sigma_{g[1]}} \tag{8}$$

$$\rho_{w_0, g[j]} = \frac{cov(w, g[j])}{\sigma_w \sigma_{g[j]}} \tag{9}$$

where $g[1]$ is the absolute value of the 1st element of the gradient vector g and $g[j]$ is the absolute value of the j th element of the gradient vector g. The ratio of the covariance and standard deviation of two variables is known as the Pearson correlation coefficient. This coefficient is used to analyze the degree to which two variables are related to one another. Then, all the obtained Pearson correlation coefficients are stored in the array $G[\,]$, and the gradient vector corresponding to the K Pearson correlation coefficients with the largest value are taken to the array $TK[\,]$, and only these K gradient vectors are uploaded to the server side. This not only decreases the cost of transmission but also prevents insignificant updates from impacting the use of processing resources and the final detection performance. Algorithm 1 is the final FedVIS pseudo-code designed in this paper.

Algorithm 1 FedVIS

B: local batch size; R:number of server-side iterations; E: number of local iterations; g: gradient vector (indexed by j) computed by the client; v: gradient value threshold; w_0: server-side initialized global gradient vector

Server global optimization :

Initialization of all server-side parameters;
for each round $t = 1, 2, \ldots R$ **do**
  Server constructs submodels by Federated dropout( ), zeroing a fixed number of activations at each fully connected layer and dropping a fixed percentage of filters at the convolution layer;
  Update all parameters to the client;
end for

Client local optimization: //execute on chosen client k

$B_k \leftarrow$(divide local client data by size B);
G=[],TK=[];/*G stores Pearson coefficients ρ_{w_0},$g[j]$ and index ind. TK stores gradient vectors */
for each local epoch i from 1 to E **do**
  Client computes the gradient vector
  for $g[j]$ in g **do**
    $G.append(j, \rho w0, g[j])$; /*Pearson coefficients ρ_{w_0},$g[j]$ according to Equation (9) */
    Sort G by $G.absValue$; /*Sort from largest to smallest. */
  end for
  for $d = 1$ to K **do**
    $TK.append(g[G[d].ind])$; /* TK's elements are gradient vectors. "ind" is the index of g. */
  end for
  Upload the gradients in TK to the server;
end for

4. Results

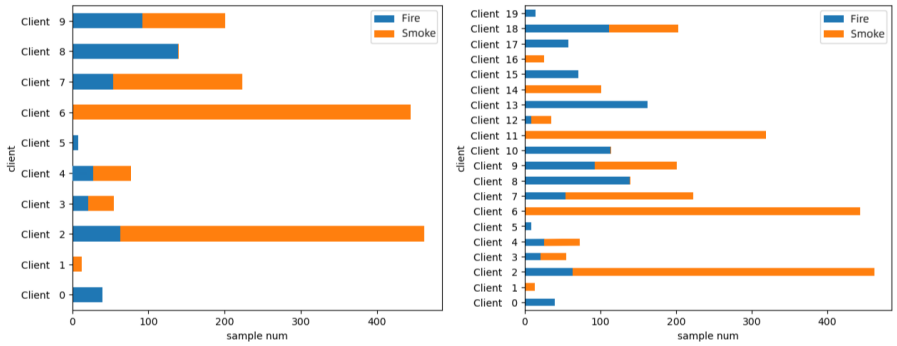
The robustness of the model and generalization to heterogeneous data are improved in this paper by intentionally introducing Gaussian noise, which aims to mimic the real federated learning environment as closely as possible. The client is a laptop with an Intel Core i58300H processor and an RPi 3 with an ARM Cortex A7 CPU, while the server is a desktop with an Intel Core i99820X Xseries Processor and an NVIDIA GeForce RTX 3060.

4.1. Dataset

In this paper, the dataset used Internet downloaded images and collected public datasets, a total of 6675 images, respectively, to detect smoke and fire. The combined dataset prepared for the work includes sources from the literature [30] and DATABASE (http://cfdb.univ-corse.fr/index.phpmenu=1 (accessed on 15 September 2022)). Example images are shown in Figure 5.

Considering the complexity of the real situation, federated learning needs to deal with various data distribution scenarios, including iid and non-iid (iid is an ideal assumption). In experiments, it is often necessary to divide the complete training dataset and test dataset according to the number of clients and the specific distribution and division strategy. The data partition is presented in Figure 6.

Figure 5. Example images. The dataset used Internet downloaded images and collected public datasets to detect smoke and fire.



(a) The number of clients is 10 (b) The number of clients is 20

Figure 6. Data Partitioner in federated learning. The dataset used Internet downloaded images and collected public datasets to detect smoke and fire.

The data splitting is performed in accordance with the k-fold cross validation mechanism. In the small sample dataset of this work, K was set to 5. Eighty percent of the dataset is used for training, while twenty percent is used for testing. The image format of the dataset is VOC. Each VOC image file correlates to an XML file with the same name, and the XML file includes the basic details of the corresponding image, including the file name, sources, file size, and information on the object area and category.

The model evaluation metrics in this paper use the average accuracy rate to evaluate all the models, mainly on the server side and client side combined.

4.2. mean Average Precision mAP

The intersection over union ratio (IoU) is used to measure the accuracy of the prediction frame and is calculated as the size of the overlap between the prediction frame and the real frame.

The mean Average Precision (mAP) is the most common metric to evaluate the performance of a model in the field of object detection, and it also takes values from 0 to 1. The following describes how mAP is calculated. First, the intersection ratio IoU of all predicted and real frames is obtained, and then the frames are sorted in the order of IoU size. Then, a threshold value for detecting overlapping regions is preset (e.g., 0.5), the precision and recall at this threshold are found, and mAP@0.5 means the sum of exact rates for all categories. The formulas for precision and recall are as follows.

$$Precision = \frac{TP}{(TP + FP)} \tag{10}$$

$$Recall = \frac{TP}{(TP + FN)} \tag{11}$$

FP stands for false positives, TP for true positives, FN for false negatives, and TN is true negatives. By averaging the Precision values corresponding to each recall value using the Precision–Recall curve as a reference, we can obtain an array evaluation: AP. The formula of AP is as follows.

$$AP = \sum_{(i=1)}^{(n-1)} (r_{(i+1)} - r_i) P_{inter}(r_i + 1) \tag{12}$$

where the recall value for the first interpolation of the precision interpolation segment, in ascending order, is r_1, r_2, r_n. The mAP (large) refers to the average accuracy of detection for large objects (962 pixels < area of the region < 100,002 pixels), and the mAP (medium) refers to the average accuracy for medium-sized objects average accuracy (322 pixels < area < 962 pixels). All AP kinds are mAP, and mAP has the following formula.

$$mAP = \frac{(\sum_{(i=1)}^{k} AP_i)}{k} \tag{13}$$

4.3. The Hyper-Parameter Tuning

In federated learning, the clients require synchronous training and parameter updates for several clients. The experimental portion of this study employs serial training for each client and uploads each client's parameters to the terminal server in order to simulate this federated learning configuration. The local model's training iterations are set to 5, and the batch size of the training data for the local model is set to 60 throughout the training phase. The learning rate is set at 0.005 by default. For each local client in this paper, the identical local model is set.

4.4. Experimental Conclusions

The coco dataset is the most common one used to conduct a preliminary test of the model. The mean Average Precision (mAP) at various IoU levels is used to assess the results. Comparing the different algorithms, it can be seen that FedVIS can achieve more than 80% accuracy after 400 epochs on the MSCOCO dataset, respectively. As shown in Table 1, we conducted experiments using different average accuracies, i.e., 0.5, 0.7, medium, and large, to test the accuracy of transformer (Centralized) and FedVIS on the MSCOCO dataset. Compared with Fedsgd, FedAvg, FedAWS, and CMFL, FedVIS was able to achieve 91.1% accuracy on the MSCOCO dataset when the average accuracy was 0.5, and transformer (Centralized) was able to achieve 93.9% accuracy on the MSCOCO dataset; when the average accuracy was 0.7, the FedVIS achieves 90.2% accuracy in the MSCOCO dataset and transformer (Centralized) achieves 91.5% accuracy in the MSCOCO dataset; when the average accuracy is medium, FedVIS achieves 88.3% accuracy in the MSCOCO dataset and transformer (Centralized) achieves 91.5% accuracy in the MSCOCO dataset; when the average accuracy is medium, FedVIS achieves 88.3% accuracy in the MSCOCO dataset and transformer (Centralized) achieves 89.1% accuracy in the MSCOCO dataset; when the average accuracy is large, FedVIS achieves 87.2% accuracy in the MSCOCO dataset and transformer(Centralized) achieves 88.2% accuracy in the MSCOCO dataset.

The above results indicate that the value of the average accuracy affects the prediction accuracy of transformer (Centralized) and FedVIS. Therefore, we conclude that lower values of average precision make transformer (Centralized) and FedVIS yield higher accuracy but provide lower accuracy of the prediction frames.

In the training of federated learning, the data are stored among different clients and the distribution of the data among clients may be various. Certain FL approaches such as FedAvg fail to solve the heterogeneity problem of the data and gradients obtained by clients

differentiate widely. As a result, the average value of gradients in the server fluctuates, which leads to the fluctuation of the algorithm's accuracy. As demonstrated in Figure 7, our method computes the Pearson coefficient between gradients in each client and these gradients in the server, and it is capable of mitigating the heterogeneity problem of the data.

Table 1. Test results of different methods on MSCOCO dataset, using the coco dataset to preliminary test the model. The results are evaluated using the mAP at different IoU thresholds. Compared to other federated learning algorithms, FedVIS has the best performance.

Method	mAP @0.5	mAP @0.75	mAP (Medium)	mAP (Large)
transformer (Centralized)	93.9	91.5	89.1	88.2
Fedsgd	88.0	73.1	62.2	75.6
FedAvg	89.2	79.1	69.0	78.6
FedAWS	90.9	82.1	71.3	79.7
CMFL	92.9	88.4	83.87	85.1
FedVIS	**91.1**	**90.2**	**88.3**	**87.2**

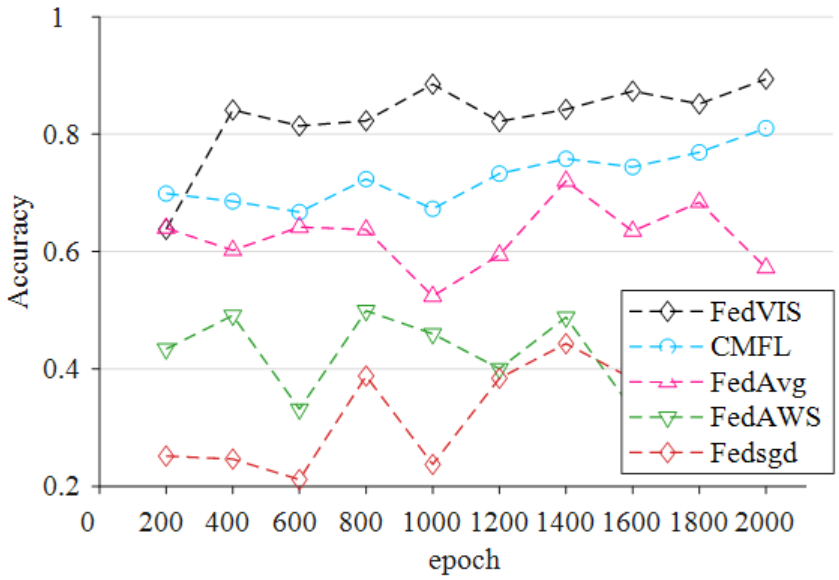


Figure 7. Test result. Accuracy vs. Epoch Graph on the testing dataset with different methods. As the epoch increases, FedVIS is at the top compared to other federation learning algorithms which indicates that FedVIS was able to learn and predict quite accurately.

4.5. Comparison Experiments

For model accuracy comparison, detection results were evaluated by calculating different threshold mean average accuracies (mAP) ranging from 0.05 to 0.95 in steps of 0.05. All experiments were repeated five times and evaluated with five different random seeds for weight initialization to ensure fairness and reasonableness, and, finally, the average of the experimental results was taken. We use three data distribution settings to compare the detection performance of different algorithmic models.

(i) IID setting, which closely resembles a real distributed learning environment. There is not a single high-performance machine hosting the model and dataset, and the training set is distributed equally among 10 clients; every single one of them builds a model and sends the model's parameters to the server for aggregation. In the IID environment, a consistent allocation of all categories will be allocated to each client. (ii) Non-IID setting, the data sorted by labels will be partitioned into 10 or 20 client partitions, and each client

will be randomly allocated just one category of data; we will then be able to assess each method's stability using non-IID data.

Table 2 and Table 3 respectively gvie the result that the algorithm is learned from IID and Non-IID distribution setttings. As one can see that our method achieves a performance comparable to that of data-centric learning methods, which is closer to 82.25%, indicating that our strategy outperforms other approaches. It is important to note that as the number of clients increases, the accuracy achieved by the other algorithmic models shows a decrease of about 2% to 5%, and the number of communication rounds required increases significantly; meanwhile, our method can still maintain a better detection performance with about the same number of communication rounds as before.

Table 2. Comparison of detection performance under IID distribution. As the number of clients increases, the accuracy achieved by the other algorithmic models shows a decrease and the number of communication rounds required increases significantly, while our method can still maintain a better detection performance.

Method	Clients	Communication Rounds (Model Convergence)	Map @0.5:0.95
transformer (Centralized)	-	-	82.25%
FedAvg	10	689	72.34%
FedAvg	20	934	71.03%
FedAWS	10	567	74.26%
FedAWS	20	657	72.43%
CMFL	10	375	78.33%
CMFL	20	499	76.25%
FedVIS	10	365	81.42%
FedVIS	20	**387**	80.32%

Table 3. Comparison of detection performance under the Non-IID distribution. As the number of clients increases, the accuracy achieved by the other algorithmic models shows a decrease and the number of communication rounds required increases significantly, while our method can still maintain a better detection performance.

Method	Clients	Communication Rounds (Model Convergence)	Map @0.5:0.95
transformer (Centralized)	-	-	82.25%
FedAvg	10	796	69.23%
FedAvg	20	1034	66.78%
FedAWS	10	663	74.26%
FedAWS	20	757	72.06%
CMFL	10	423	77.21%
CMFL	20	567	75.29%
FedVIS	10	414	80.06%
FedVIS	20	**443**	79.34%

Specifically, Table 4 lists the communication rounds required by FedAVG algorithm, CMFL algorithm, and FedVIS algorithm to reach the target accuracy at different correlation

thresholds. Among them, "The average number of communication rounds decreased compared to FedAvg" should be the average number of communication rounds decreased compared to FedAvg algorithm when FedVIS algorithm achieves different target accuracy rates. For example, according to the data in Table 4, in the line of FedVIS, the specific calculation process of "the average number of communication rounds decreased compared to FedAvg" is [(50–39)/50 + (130–76)/130 + (330–248)/330 + (960–840)/960]/4 = 54.67%. Similarly, "the average number of communication rounds decreased compared to FedAvg" and compared with CMFL, the average number of communication rounds is reduced. Because FedVIS uses the federated dropout algorithm and gradient selection algorithm to reduce the communication overhead, FedVIS discards some parameters in the communication process, making it possible to reduce the number of communication rounds by about 25% and 9%, respectively, on the premise of ensuring accuracy.

Table 4. The number of communication rounds required by FedAvg algorithm, CMFL algorithm, and FedVIS algorithm to reach the target accuracy at different correlation thresholds. The average number of communication rounds decreased compared with FedAvg, and the average number of communication rounds decreased compared with CMFL.

Algorithm	**Communication Rounds**				**Compared with FedAvg**	**Compared with CMFL**
	40%	50%	60%	70%		
FedAvg	50	130	330	960		
CMFL	44	100	264	886		
FedVIS	39	76	248	840	25.22%	9.52%

4.6. The Ablation Study

In order to verify that FedVIS algorithm improves the prediction accuracy of federated learning, it is not only related to the federated dropout and gradient selection algorithm but also related to the replacement of traditional neural network by transformer. By separating the different steps of the algorithm and performing ablation experiments, compared with the classical federal average model, the effectiveness of FedVIS is verified. The experimental results are listed in the Table 5.

Table 5. The ablation study. The effectiveness of FedVIS was verified by ablation experiments through different steps of separating FedVIS, compared with the classical FedAvg.

Algorithm	**CNN**	**Transformer**
FedAvg	97.05	98.8
FedVIS	**98.46**	**99.98**

4.7. Comparison of Model Generalizability in Complex Scenarios

As shown in the Figure 8, based on the basic transformer model, we compare the detection effects under different federated learning methods and deliberately choose different detection scenarios from the training set to compare FedAvg, CMFL, and the FedVIS method in this paper. The intuitive experimental results show that in the complex and occluded detection scenes, the method in this paper not only obtains more accurate detection frames but also can retain as many accurate detection results as possible, i.e., with higher accuracy and recall.

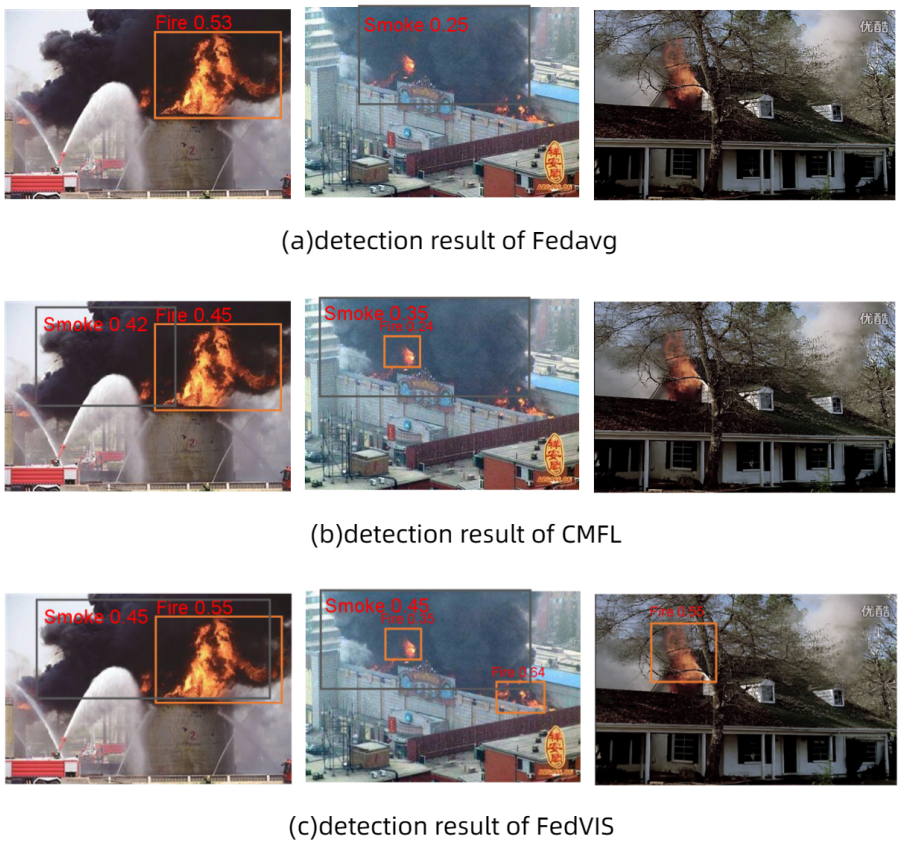


Figure 8. Comparison of the actual detection effect of FedAvg, CMFL, and FedVIS. The figure (**a**) shows the detection result of FedAvg. The figure (**b**) shows the detection result of CMFL. The figure (**c**) shows the detection result of FedVIS. For the detection of visible flames and smoke, FedVIS performs best. For the detection of visible smoke and inconspicuous flames, FedVIS performs best. For the detection of obscured flames, only FedVIS detects the flame.

5. Conclusions

The FedVIS algorithm designed in this paper improves the efficiency of fire detection by reducing network bandwidth consumption and model training time compared with the traditional centralized learning method through a modified federated learning approach. It also greatly reduces network model parameters and computation on the client and server sides compared with other federated learning algorithms. In addition, replacing traditional neural networks with transformer models in federated learning effectively boosts the model's robustness to heterogeneous data from fire monitoring equipment, resulting in a more accurate fire detection and localization model.

Author Contributions: Y.H.: methodology, data curation, writing—original draft preparation. X.F.: conceptualization, visualization, software. W.Z.: supervision, writing—review and editing, investigation. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Natural Science Foundation of China No. 61872062.

Data Availability Statement: In this paper, the dataset used Internet downloaded images and collected public datasets, a total of 6675 images, respectively, to detect smoke and fire. The synthetic dataset prepared for the work includes sources from the literature [30] and the CORSICAN FIRE DATABASE (http://cfdb.univ-corse.fr/index.phpmenu=1 (accessed on 15 September 2022)).

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

CNN	Convolutional Neural Network
IoT	Internet of Things
NIR	Near Infrared
NLP	natural language processing
RPi	Raspberry Pi
VOC	Visual Object Classes
mAP	mean Average Precision
IoU	Intersection over Union
TP	True Positives
FP	False Positives
FN	False Negatives
IID	Independent and Identically Distributed
Non-IID	Not Independent and Identically Distributed

References

1. Fire and Rescue Bureau, Ministry of Emergency Management. There Were 219,000 Fires Nationwide in the First Quarter, with 625 Deaths [EB/OL].(2022-04-04)[2022-5-26]. Available online: https://www.119.gov.cn/articlel47Sd3LDSJTA (accessed on 15 September 2022).
2. Mothukuri, V.; Parizi, R.M.; Pouriyeh, S.; Huang, Y.; Dehghantanha, A.; Srivastava, G. A survey on security and privacy of federated learning. *Future Gener. Comput. Syst.* **2021**, *115*, 619–640. [CrossRef]
3. Mueller, M.; Karasev, P.; Kolesov, I.; Tannenbaum, A. Optical flow estimation for flame detection in videos. *IEEE Trans. Image Process.* **2013**, *22*, 2786–2797. [CrossRef]
4. Foggia, P.; Saggese, A.; Vento, M. Real-time fire detection for video- surveillance applications using a combination of experts based on color, shape, and motion. *IEEE Trans. Circuits Syst. Video Technol.* **2015**, *25*, 1545–1556 [CrossRef]
5. Rajpoot, Q.M.; Jensen, C.D. Video surveillance: Privacy issues and legal compliance. In *Promoting Social Change and Democracy through Information Technology*; IGI Global: Hershey, PA, USA, 2015; pp. 69–92.
6. Muhammad, K.; Ahmad, J.; Baik, S.W. Early fire detection using convolutional neural networks during surveillance for effective disaster management. *Neurocomputing* **2018**, *288*, 30–42. [CrossRef]
7. Muhammad, K.; Ahmad, J.; Mehmood, I.; Rho, S.; Baik, S.W. Convolutional neural networks based fire detection in surveillance videos. *IEEE Access* **2018**, *6*, 18174–18183. [CrossRef]
8. Muhammad, K.; Khan, S.; Elhoseny, M.; Ahmed, S.H.; Baik, S.W. Efficient fire detection for uncertain surveillance environment. *IEEE Trans. Ind. Inform.* **2019**, *15*, 3113–3122. [CrossRef]
9. Khan, Z.A.; Hussain, T.; Ullah, F.U.M.; Gupta, S.K.; Lee, M.Y.; Baik, S.W.Randomly Initialized CNN with Densely Connected Stacked Autoencoder for Efficient Fire Detection. *Eng. Appl. Artif. Intell.* **2022**, *116*, 105403. [CrossRef]
10. Xu, R.; Lin, H.; Lu, K.; Cao, L.; Liu, Y. A forest fire detection system based on ensemble learning. *Forests* **2021**, **12**, 217. [CrossRef]
11. Liu, S.; Kong, L.; Wang, H. Face detection and encryption for privacy preserving in surveillance video. In *Proceedings of the Pattern Recognition and Computer Vision: First Chinese Conference, PRCV 2018, Guangzhou, China, 23–26 November 2018*; Springer: Cham, Switzerland, 2018; pp. 162–172.
12. Luo, Z.; Hsieh, J.T.; Balachandar, N.; Yeung, S.; Pusiol, G.; Luxenberg, J.; Fei-Fei, L. Computer vision-based descriptive analytics of seniors' daily activities for long-term health monitoring. *Mach. Learn. Healthc.* **2018**, *2*, 1–18.
13. Jain, A.; Srivastava, A. Privacy-preserving efficient fire detection system for indoor surveillance. *IEEE Trans. Ind. Inform.* **2022**, *18*, 3043–3054. [CrossRef]
14. Mcmahan, H.B.; Moore, E.; Ramage, D.; Hampson, S.; Arcas, B. Communication-efficient learning of deep networks from decentralized data. In Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, Fort Lauderdale, FL, USA, 20–22 April 2017; pp. 1273–1282.
15. KhoKhar, F.A.; Shah, J.H.; Khan, M.A.; Sharif, M.; Tariq, U.; Kadry, S. A review on federated learning towards image processing. *Comput. Electr. Eng.* **2022**, *99*, 107818. [CrossRef]
16. Das, A.; Brunschwiler, T. Privacy is what we care about: Experimental investigation of federated learning on edge devices. In *Proceedings of the First International Workshop on Challenges in Artificial Intelligence and Machine Learning for Internet of Things, ser. AIChallengeIoT'19*; Association for Computing Machinery: New York, NY, USA, 2019; pp. 39–42.
17. Mcmahan, H.B.; Moore, E.; Ramage, D.; Arcas, B.A.Y. Federated Learning of Deep Networks using Model Averaging. In Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS), Lauderdale, FL, USA, 20–22 April 2017.
18. McMahan, H.; Moore, E.; Ramage, D.; Arcas, B. Federated learning of deep networks using model averaging. *arXiv* **2016**, arXiv:1602.05629.

19. Wang, L.; Wang, W.; Li, B. CMFL: Mitigating Communication Overhead for Federated Learning. In Proceedings of the 2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS), Dallas, TX, USA, 7–9 July 2019; pp. 954–964.
20. Rawat, A.S.; Yu, X.; Menon, A.K.; Kumar, S. Federated learning with only positive labels. *arXiv* **2020**, arXiv:2004.10342.
21. Majid, S.; Alenezi, F.; Masood, S.; Ahmad, M.; Gunduz, E.S.; Polat, K. Attention based cnn model for fire detection and localization in real-world images. *Expert Syst. Appl.* **2022**, *189*, 116114. [CrossRef]
22. Wu, Q.; He, K.; Chen, X. Personalized federated learning for intelligent iot applications: A cloud-edge based framework. *arXiv* **2020**, arXiv:2002.10671.
23. Qu, L.; Zhou, Y.; Liang, P.P.; Xia, Y.; Wang, F.; Adeli, E.; Fei-Fei, L.; Rubin, D. Rethinking architecture design for tackling data heterogeneity in federated learning. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 10051–10061.
24. Tao, J.; Gao, Z.; Guo, Z. Training Vision Transformers in Federated Learning with Limited Edge-Device Resources. *Electronics* **2022**, *11*, 2638. [CrossRef]
25. Zhou, D.; Yu, Z.; Xie, E.; Xiao, C.; Anandkumar, A.; Feng, J.; Alvarez, J.M. Understanding the Robustness in Vision Transformers. In Proceedings of the 39th International Conference on Machine Learning PMLR, Baltimore, MD, USA, 17–23 July 2022.
26. Qiu, X.; Ye, Z.; Cui, X.; Gao, Z. Survey of communication overhead of federated learning. *J. Comput. Appl.* **2022**, *42*, 333–342.
27. Li, T.; Sahu, A.K.; Zaheer, M.; Sanjabi, M.; Talwalkar, A.; Smith, V. Federated optimization in heterogeneous networks. *arXiv* **2018**, arXiv:1812.06127.
28. Caldas, S.; Koneny, J.; McMahan, H.B.; Talwalkar, A. Expanding the reach of federated learning by reducing client resource requirements. *arXiv* **2018**, arXiv:1812.07210.
29. Chen, T.; Sun, Y.; Yin, W. Communication-adaptive stochastic gradient methods for distributed learning. *IEEE Trans. Signal Process.* **2021**, *69*, 4637–4651. [CrossRef]
30. Ma, S.; Nie, J.; Kang, J.; Lyu, L.; Liu, R.W.; Zhao, R.; Liu, Z.; Niyato, D. Privacy-preserving anomaly detection in cloud manufacturing via federated transformer. *IEEE Trans. Ind. Inform.* **2022**, *18*, 8977–8987. [CrossRef]