Multistage Real-Time Fire Detection Using Convolutional Neural Networks and Long Short-Term Memory Networks

MANH DUNG NGUYEN, HOAI NAM VU, DUC CUONG PHAM, BOKGIL CHOI, AND SOONGHWAN RO

1Department of Electronic Engineering, Posts and Telecommunications Institute of Technology, Hanoi 12110, Vietnam
2Department of Computer Science, Posts and Telecommunications Institute of Technology, Hanoi 12110, Vietnam
3IVS Vietnam Company, Hanoi 12110, Vietnam
4Department of Electrical Engineering, Kongju National University, Cheonan 31080, South Korea
5Department of Information and Communication, Kongju National University, Cheonan 31080, South Korea

Corresponding author: Soonghwan Ro (rosh@kongju.ac.kr)

This work was supported in part by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education under Grant NRF2018R1D1A1B07049146. The work of Hoai Nam Vu was supported by Vingroup Joint Stock Company and by the Domestic Master/Ph.D. Scholarship Programme of Vingroup Innovation Foundation (VINIF), Vingroup Big Data Institute (VINBIGDATA), under Grant VINIF.2020.TS.103.

ABSTRACT Fire is one of the most commonly occurring disasters and is the main cause of catastrophic personal injury and devastating property damage. An early detection system is necessary to prevent fires from spreading out of control. In this paper, we propose a multistage fire detection method using convolutional neural networks (CNNs) and long short-term memory (LSTM) networks. In the first stage, fire candidates are detected by using their salient features, such as their color, flickering frequency, and brightness. In the second stage, a pretrained CNN model is used to extract the 2D features of flames that are the input for the LSTM network. In the last stage, a softmax classifier is utilized to determine whether the flames represent a true fire or a nonfire moving object. The experimental results show that our proposed method can achieve competitive performance compared with other state-of-the-art methods and is suitable for real-world applications.

INDEX TERMS Fire detection, convolutional neural network, ImageNet, long short-term memory.

I. INTRODUCTION

We frequently hear about fires in the news. Every year, fires cause thousands of human deaths and billions of dollars in property damage. Fire monitoring and protection are always key concerns when managing apartment buildings, warehouses, forests, substations, railways, and tunnels. If fires are not detected early and become out of control, the consequences are often disastrous. Developing a system that is able to automatically detect fire at an early stage is necessary for protecting both human life and property.

The first generation of fire detection systems use sensors, such as ionization detectors, photoelectric sensors and carbon dioxide detectors. Although these systems have had some success in detecting fire, sensor-based systems still have many limitations, especially in large, open areas. Each sensor monitors a narrow region, so a fire cannot be detected immediately if it is far from a sensor. Furthermore, the accuracy of this methodology depends heavily on the sensor density and sensor reliability, which leads to some inconvenience in terms of the cost and installation. Sensor-based fire detection is widely used; however, missed detections of real fires and a high false detection rate are common.

Digital cameras, namely, closed-circuit television (CCTV), have been rapidly evolving in the field of security surveillance. Compared to sensor-based systems, security cameras are easy to install and can be used to monitor large, open areas. Currently, CCTV is everywhere, and utilizing a CCTV system for monitoring fire may be an economical and efficient solution. Recently, sensor-based systems have started to be replaced by surveillance cameras and video analysis systems. A large number of image processing algorithms have been proposed for smoke and fire detection via video analysis, and some of them have achieved considerable success. Although computer vision-based systems have been
shown to have advantages over sensor-based systems, their performance and results are still far from ideal.

Machine learning (ML) algorithms are algorithms that automatically build a mathematical model using sample data, also known as training data, to make decisions without being specifically programmed to make those decisions. Machine learning has been developing since the 1950s; however, initially, the archived results were not very impressive. The fundamental difficulties faced during this period in the development of machine learning were data collection and the limitations of computing resources. However, over the last decade, as computers have become faster and especially due to the explosion of the internet, data collection has become easier than ever, and the development of ML is rapidly progressing. Recently, a new branch of ML called deep learning has emerged as a state-of-the-art machine learning algorithm. These algorithms have shown extremely good performance for computer vision applications, including object detection and image classification. This significant change provides an opportunity to solve many problems that still exist in computer vision, including fire detection.

This paper proposes a method to process fire videos using a multistage CNN-LSTM model. The proposed method consists of a CNN model and an LSTM model that are utilized to extract spatial and temporal features, respectively, to effectively detect fire in videos. The main contributions of the proposed method are as follows:
- We have proposed a method that integrates fire candidate extraction and CNN-LSTM classification for detecting fire in videos.
- The proposed method can detect fire with high accuracy while maintaining a low false detection rate.
- The proposed method can detect fire at different scales and environmental conditions.
- The proposed method is fast; therefore, it can be integrated into real-world applications.
- We have collected a fire video dataset for training and testing the algorithm. We have also made it public for future research purposes.

The remainder of this paper is organized as follows. Section 2 reviews the related works. Section 3 describes the proposed system. Section 4 reports the experimental results. Future research directions and a discussion are provided in Section 5.

II. RELATED WORK
In the literature, many approaches have been proposed for early fire detection based on video surveillance. The workflows of these approaches are mainly separated into the following three substeps: fire-colored pixel detection, moving pixel detection, feature extraction and classification.

From observation, we can easily determine that the flame color ranges from red–yellow to white depending on the temperature and material. Flame color is a very important component of fire detection in many algorithms; there are many proposed approaches that rely on modeling fire-colored pixels. These approaches can be roughly classified into the following three categories: fire color rule-based methods, polynomial-based color models and Gaussian distribution-based models. The color rules require several heuristic rules, which are based on the RGB (Red - Green - Blue) [1]–[6], Ycbcr [7]–[9], YUV [10] or Lab [11] color space. The fire region is manually segmented, and the relationship between the pixel values of the three channels is analyzed to estimate a rule to classify the fire-colored pixels in an image. The RGB distribution of the fire flame pixels in images is a two-dimensional representation of the flame pixel color. Different color distributions and flame pixel processes can be obtained by observing different types of fires, depending on the burning material. Based on this distribution, several heuristic rules combined with decision thresholds are estimated to classify the fire-colored pixels in an image. Although different methods are used to model the color of flames, most of these approaches have achieved good results when detecting fire-colored pixels in images. However, in real-world situations, many objects may be similarly colored; therefore, if we use only the color information, false detections could occur. Therefore, further steps are needed to eliminate non-fire-colored objects.

In addition to color, motion is also an important feature for recognizing fire. Background subtraction is commonly used to segment out the moving objects in a scene for surveillance applications. There are several methods [2]–[4], [6], [11]–[13] that treat fire as a moving object with the assumption that the appearance of fire will change the background; therefore, these methods use background subtraction as the first step when segmenting flames. Almost all of the presented papers in the state-of-the-art literature use either frame difference or background-based methods to detect nonstationary pixels. The candidate fire pixels are detected by combining the color and motion results detected previously. However, these methods cannot distinguish fire-colored moving objects from fire. Therefore, additional steps are required to accurately extract flames from a video sequence.

In the literature, flickers are also a widely used feature for detecting fire. We can easily observe that the luminance of fire randomly changes over time. Based on this feature, the author of [12] proposed a 1-D wavelet transform for temporal color variation analysis. Wavelet signals can easily reveal the random characteristic of a given signal, which is an intrinsic feature of flame pixels. Similarly, [1] proposed an algorithm that uses the flicker characteristic to detect fire. This algorithm calculates the cumulative time derivative of luminance because of the tendency of fire to periodically flicker around a region, thus giving the flickering regions of the fire the strongest values. The author of [4] analyzed the flame flickering frequency and discovered that the frequency was approximately 10 Hz; therefore, they verified the candidate flame pixels using a hidden Markov model (HMM) to determine whether their frequencies were approximately 10 Hz. Ref. [13] extracted the normalized red skewness, LH wavelet coefficient skewness, HL wavelet coefficient skewness, and
HH wavelet coefficient skewness from candidate flame pixels. Then, they applied Bayesian networks to estimate the probability that the current frame contains fire.

Other algorithms have been used to extract features from candidate pixels, such as the position and area, wavelet information, boundary and flickering frequency, and the temporal and spatial variation of the intensity. Probabilistic models are estimated based on these features, and heuristic rules are applied to distinguish between true fires and nonfire moving objects. These algorithms may have good detection rates; however, their false detection rates are still too high to meet the requirements of security applications. Probabilistic models have a major shortcoming in that it is difficult to determine the optimal thresholds for classification. A lenient threshold could result in many false-positives while a strict threshold could lead to many missed fires.

Reducing false fire detection is a challenge, and increasingly more studies are attempting to minimize false detections. Recently, some image classification methods using machine learning algorithms have been developed, which may be a good approach for distinguishing true fires from nonfires.

Deep learning is a subset of machine learning [14]–[17], [18] that achieves great power and flexibility by learning to represent the world as a nested hierarchy of concepts, where each concept is defined in relation to simpler concepts and more abstract representations are computed in terms of less abstract representations. A deep learning technique learns categories incrementally using its hidden layer architecture and defines low-level categories. Deep learning promises to provide more accurate machine learning algorithms compared to traditional machine learning algorithms, which have little to no feature engineering. Deep learning algorithms based on CNNs have emerged as a state-of-the-art technique in image classification and object recognition. These algorithms have achieved extremely good performance in computer vision applications, including image classification and object detection. Many experiments show that CNNs are very good for image classification and that applying deep learning algorithms to fire classification may be able to reduce false detections. A recent trend uses the advantage of CNNs to improve the accuracy of fire detection, and some methods using this strategy have achieved considerable success. Ref. [19] investigated the problem of fire detection in the real world by creating an unbalanced dataset. The author stated that fire is a rare event in real-world conditions that leads to the failure of many state-of-the-art CNN models. They proposed a deeper CNN model that achieved promising results in detecting fire in videos to address this problem. The author of [20] applied a CNN image classifier [14] at the last layer of a cascade classification model to distinguish fire-like moving objects from real fires. The precision of CNN image classification is impressive compared to previous research methods. However, CNNs are only good at modeling an object’s 2D texture and ignore the temporal characteristics of fire flames. In fact, there are many cases in which human eyes are unable to distinguish different objects with a single image; instead, we need to observe the dynamic texture of an object over a period of time to reach a final conclusion. Ref. [21] used the extreme learning machine classifier after the CNN model to detect fire in video. Their method is able to outperform state-of-the-art deep CNNs in fire detection accuracy while maintaining a fast processing time. In [22], the author tried to detect fire directly by using advanced CNN object detection models such as YOLO, Faster-RCNN, R-FCN, and SSD. However, these algorithms are also based on 2D textures and do not consider temporal information. Furthermore, fire flames exhibit an amorphous shape, and this shape leads to difficulties when making training datasets for training a deep learning fire flame detection model. Ref. [12] proposed a 1-D wavelet transform for temporal color variation analysis of fire flames, and [23] also proposed a randomness test model to verify the dynamic textures of fire flames. However, the general disadvantage of these probabilistic methodologies is that it is difficult to determine the optimal decision threshold; furthermore, handcrafted analysis features require highly skilled expertise.

An LSTM network [24], [25] is constructed by looping the LSTM units, where each LSTM unit consists of a memory cell and three multiplicative components (an input gate, an output gate, and a forget gate). Each memory cell (or LSTM unit) uses four neural network layers, instead of one, as the RNN unit. In addition, the output of one neural network layer, in which a sigmoid function is applied, is used to control the cell state from the past cell. Because the output of the sigmoid function is in the range of 0 to 1, the past cell state is “forgotten” if the output of the sigmoid function is zero. Otherwise, it accumulates in the cell after multiplication by the controlling factor. Using this method of manipulation, LSTM is able to remember information for a long period of time. While CNNs and handcrafted feature probabilistic models have limitations when modeling dynamic textures, recurrent neural networks (RNNs) and long short-term memory (LSTM) networks have achieved great success when processing sequential multimedia data and have yielded state-of-the-art results in speech recognition, digital signal processing, video processing, and text data analysis. In [26], CNN-LSTM was used to detect the novel coronavirus (COVID-19) using X-ray images. The CNN is used to extract the potential features that are passed to the LSTM model to obtain the final results. Their proposed method can achieve an accuracy of 99.4% and notably achieved a sensitivity of 99.3% for detecting the appearance of COVID-19. Similarly, the author of [27] proposed a combination of a CNN and an RNN to solve the same problem of detecting COVID-19 from X-ray images. They used a pretrained VGG-19 model as the backbone to extract spatial features and an RNN to extract temporal features. In addition, they also used gradient-weighted class activation mapping (Grad-CAM) to visualize the region responsible for causing COVID-19 in X-ray images. In order to solve foreground segmentation, the author of [28] used a combination of a 3D CNN and
LSTM. They modeled the foreground-background segmentation problem as an encoder-decoder. The results show that their proposed algorithm achieved competitive performance in terms of the figure of merit evaluated compared to state-of-the-art methods. Inspired by the success of LSTM for sequential data analysis, Amin Ullah proposed a model that combines a CNN and LSTM to analyze the dynamic textures of image sequences [24]. This model inherits the advances of both the CNN and the LSTM network to achieve significant results for real-time action recognition in video. CNNs can
act as feature extractors for each image in a sequence while LSTM acts as sequence analysis to predict human activities. Since the CNN-LSTM combination works well for human activity recognition, the method is also well suited for analyzing the dynamic features of fire flames. In the literature, many studies have applied this model to enhance the accuracy of fire flame detection and have achieved good results. The author of [29] added CNN layers to the front end, followed by LSTM layers with a dense layer on the output. The CNN layer works as a feature extractor, and LSTM works as a video classifier for fire and nonfire classification. Similarly, the author of [30] also proposed a CNN-LSTM model for fire image sequence classification. While [29] directly inputs raw images into the CNN input layer, the author uses optical flow to transform raw images into motion images before the images are input to CNNs for feature extraction. Although both of the abovementioned algorithms have reported good results, there is no preprocessing step to localize the position of fire flames; therefore, these algorithms are only suited for large fires in which fire flames are a major part of the image. The prediction accuracy may decrease when the flame is small and occupies a small area in the scene.

III. MULTISTAGE FIRE FLAME DETECTION

The proposed fire flame detection algorithm is illustrated in Figure 1. This approach is mainly separated into three stages. In the first stage, candidate fire flames, which are very likely real fires, will be detected and localized by using the color and motion features. Our algorithm can detect multiple fire candidates in an image and track the fire candidates in the next frames. This approach can ensure that no real fire is missing from the video. In the next stage, a cropped image of each candidate from sequence images will be input into the CNN model and then transformed into visual features through CNN layers. Finally, the extracted fire flame features in the image sequence are passed to the LSTM model, and the LSTM output predicts whether the image sequence contains a real fire or a nonmoving object.

A. DETECTING CANDIDATE FIRE REGIONS

In the first step, we seek to localize the candidate fire regions, i.e., the regions where there is a high possibility of apparent fire flames. Only color and motion characteristics are used to detect fire flames at this stage. Color is a distinct feature of fire flames; commonly, flame color varies from yellow to red to white, depending on the fire temperature. To simulate the color sensing properties of the human visual system, RGB color information is usually transformed into a mathematical space that decouples brightness (or luminance) information from color information.

Among these color models, the HSV (Hue - Saturation - Value of intensity) color model is suitable for providing a more human-oriented method for describing colors. Therefore, HSV is very convenient for color analysis.

A useful fire color model that uses HSV information was previously introduced in [2]. We slightly modify this model for our fire pixel classification, in which an image pixel can be classified as a fire pixel if it meets the following conditions:

\[ H_{T_1} \leq H \leq H_{T_2}, \]
\[ R > R_T, \]
\[ S > \frac{(255 - R) \times S_T}{R_T}, \]

where \( H, R, \) and \( S \) are the hue, intensity of the red channel, and saturation of the image pixels, respectively. \( H_{T_1}, H_{T_2}, R_T \) and \( S_T \) are defined through intensive experiments. \( H_{T_1} \) and \( H_{T_2} \) usually range from 0 to 70, \( S_T \) is approximately 60 and \( R_T \) is approximately 120.

Figure 2 displays the results of fire-colored pixel classification. We can see that many background image pixels are incorrectly classified as fire-colored. Therefore, color is insufficient to detect fire, and further processing steps are needed to eliminate stationary background image pixels that are fire-colored. In addition to color, another useful feature is flame flickering, which can be used to improve fire pixel classification. Because of flickering, the intensity of both flame pixels and the surrounding pixels tends to change randomly over time. Many previous approaches have attempted to detect the fire pixel flicker frequency; however, this frequency is very difficult to measure.

The author of [10] developed an effective method for detecting fires according to the frequency of luminance flicker. This algorithm calculates the cumulative time derivative of the luminance. Because of fire’s tendency to periodically flicker around a region, the flickering fire regions have the strongest values.

The time derivative of the luminance is zero for stationary regions and nonzero for moving objects. Thus, the time derivative for the video images is able to track a moving object. The sum of the absolute values of the derivatives increases if the object moves periodically around a region. In a fire scene, the flickering of the fire permanently increases the pixel values near the fire region. Following this method, we can estimate the cumulative fire flicker energy map as follows:

\[ E(t) = I(t) - I(t - 1), \]
\[ A(t) = \alpha E(t - 1) + (1 - \alpha)E(t), \]

where \( I(t) \) and \( E(t) \) are the intensity and flicker energy of the image pixels at time \( t \), respectively. \( A(t) \) and \( \alpha \) are the cumulative flicker energy and a cumulative factor, respectively.
Figure 3 shows the calculation of the cumulative time derivative for the fire image pixel intensity. Brighter areas have a higher cumulative flicker energy. For surrounding flames, the cumulative flicker energy is highest.

Once both the color map and the flicker energy map are estimated for the fire pixels, we can use per-pixel operations to fuse the color map and flicker map into a possible fire pixel map. Figure 4 illustrates the steps involved in possible fire pixel map segmentation. From top to bottom, Figure 4 includes the original image, color map, flickering energy map, fusion map, and refined possible fire pixel map, respectively. The final step extracts fire pixels to apply morphological transformations, such as eroding, dilating, opening or closing functions, to the fusion binary map to refine the map.

The candidate fire regions can be segmented from possible fire pixel maps. Figure 5 shows an example of detecting candidate fire regions, where a is the original image; b is the fire-colored pixel map; c is the flickering energy map; d is the fusion map; e is the refined map; and f is a candidate fire region in the subimage, shown by the green rectangle. Multiple candidate fire regions can be detected concurrently, and regions that are near each other will be merged into one region.

Per-pixel level analysis is insufficient for removing moving objects that resemble fire. Therefore, further processes at high spatial analysis levels are required. To ensure that all connected possible fire pixels are clustered into subregions as candidate fire regions, each candidate fire region is treated
as a moving object. Finally, the detected moving objects are tracked and classified to eliminate nonfire moving objects.

B. CLASSIFYING FIRE CANDIDATE USING CNN-BiLSTM

As analyzed, the previous methods that predict fires based on a single image are insufficient, and we need to observe objects in sequence to make decisions. For real fire/nonfire classification, we use a CNN-BiLSTM model. The CNN-BiLSTM architecture involves using CNN layers for extracting features from input data, combined with BiLSTM to support sequence prediction as shown in Figure 7.

The network architecture selected for feature extraction in this paper is based on the ResNet-18 architecture [15], which represents a perfect trade-off between model complexity and performance. The network consists of five convolutional stages as demonstrated in Figure 6. The network constructs a hierarchical representation of input images in which deeper layers contain higher-level features constructed using the lower-level features from earlier layers. To obtain feature representations from the training and test images, activation functions are used in the global pooling layer, ‘pool5’, at the end of the network. The global pooling layer pools the input features over all spatial locations, producing 256 features in total.

The sequence images from each candidate fire region will be passed through CNNs for visual feature extraction. Then, these independent features are fed into a many-to-one multilayer BiLSTM network that temporarily fuses this extracted information. Figure 7 describes the architecture of the CNN-BiLSTM model for fire/nonfire classification. We use a multilayer BiLSTM model to boost the network performance. The architecture includes two BiLSTM layers stacked on top of each other: one LTSM moves in the forward direction and the other moves in the reverse direction. Their combined output is then computed based on the hidden state of both layers. The softmax classifier is applied during the final state of LSTM to make the final decision.

The operation of the LSTM network is explained in Figure 8. The left side is the LSTM unit, and the right side is the multilayer LSTM unit. \( x_t \) is the image feature extracted by the CNNs, and it is also the input to the LSTM unit at time \( t \). First, LSTM determines which information from the previous cell memory state \( c_{t-1} \) can be forgotten at forget gate \( f_t \). The input to \( f_t \) is \( x_t \), and the previous time step’s hidden state \( h_{t-1} \). The output is a number in the range \([0, 1]\) for each state of \( C_t \). If the output is 1, all information will be saved; however, if the output is 0, all information will be forgotten. Equation (6) illustrates this step.

The next step is to select which information \( i_t \) can be stored in the cell memory states at the input gate layer. This layer is a sigmoid layer that determines which information should be updated. The next layer is the tanh layer, where a new value \( \tilde{c}_t \) is added to the cell states. Then, the previous cell state \( (c_{t-1}) \) will be updated to the new state \( c_t \). The previous cell memory state is multiplied by \( f_t \) to clear information that the unit determined should be forgotten in the previous step. Next, \( i_t \ast \tilde{c}_t \) is added to produce new memory cell states. The principles behind this step are described in Equations (7, 8, and 9).

In the final step, we determine which information is needed at the output, and the value of the output depends on the state of the memory cell. A sigmoid layer is applied to determine which part of the information should be output. The final decision is made by applying a softmax classifier to the final LSTM state to classify real fires from nonfire moving objects.
This step is demonstrated in Equations (10, 11, and 12).

\[ f_t = \sigma(W_f[h_{t-1}, x_t] + b_f), \]  
\[ i_t = \sigma(W_i[h_{t-1}, x_t] + b_i), \]  
\[ \hat{c}_t = \tanh(W_C[h_{t-1}, x_t] + b_C), \]  
\[ C_t = f_t \times C_{t-1} + i_t \times \hat{c}_t, \]  
\[ o_t = \sigma(W_o[h_{t-1}, x_t] + b_o), \]  
\[ h_t = o_t \times \tanh(C_t), \]  
\[ \text{Fire} = \text{softmax}(h_t). \]  

The performance of the deep neural network has been boosted by increasing the number of layers in the neural network models. Similarly, in our network, we stacked two LSTM layers to improve the system accuracy. Figure 8 shows the architecture of the two-layer LSTM network used in our algorithms. Layer 1 obtains input \( x_t \) from CNN feature extraction, and the input to layer 2 comes from the current time step \( h_{t-1} \) and the output of the current time step of layer one \( h_1 \). We also use a bidirectional LSTM model, where the output at time \( t \) is dependent not only on the previous frames in the sequence but also on the upcoming frames. The bidirectional LSTM used here is quite simple and includes two LSTMs stacked on top of each other. One LSTM moves in the forward direction, and the other moves in the backward direction.

**C. TRAINING CNN MODEL**

To train the CNN model, we use a transfer learning strategy. We fine-tune a pretrained ResNet18 model that was well trained on the ImageNet dataset with millions of images from 1000 object categories. Although the ImageNet dataset is very large, there is little fire flame and fire-like moving object data. Retraining the CNN model on a new dataset allows deeper study of the visual features of fire flames, and fire-like moving objects are necessary for high-quality training. Because the original network was well trained on a large dataset, instead of retraining the network from scratch, transfer learning utilizes a model trained on a different dataset and adapts it to train a new classifier model by continuing to back-propagate the trained model weights with a new image dataset. Transfer learning helps improve model accuracy, reduce training time and solve overfitting problems while training models on small datasets. The training dataset includes a total of 30,000 fire images and 30,000 nonfire images in different scenarios. The nonfire object images were collected from various sources, such as the PETA dataset (pedestrian images), the Cars dataset (vehicles), and the PASCAL dataset (backgrounds and other moving objects). We also manually segmented nonfire objects from surveillance videos that we ourselves recorded or downloaded from the internet. The fire object images from the videos we collected were manually segmented. Here, 60% of the dataset was used for training, 20% was used for validation and 20% was used for testing. Figure 10 shows examples of non-fire and fire training images.

We began the fine-tuning process with a learning rate of 0.01 and decreased it by a factor of ten every 2,000 iterations.
tions. We used a smaller learning rate for weights that were fine-tuned under the assumption that the pretrained CNN weights were relatively good. We did not want to distort them too quickly or by too much. The optimization process was run for a maximum of 50,000 iterations. The accuracy of the trained fire classification model was 97.5%, the false negative rate was 3.3%, and the false-positive rate was 2.7%. Figure 9 shows the training process. A learning curve visualizes the training and validation losses and validation accuracy according to the number of iterations. The learning curve shows that in the first 2000 iterations, the model accuracy improved quickly and reached a limitation at 10,000 iterations. Then, the accuracy of the model did not change significantly.

D. TRAINING THE LSTM MODEL
We implemented a bidirectional LSTM network with 16 time steps and 512 hidden units for fire/nonfire sequence image classification. The frame rate of our video dataset is reduced to 10 FPS (Frame per Second) to reflect the fact that the change in flame shape is nonsignificant between consecutive frames in the original videos. The visual features of a flame candidate over 16 consecutive frames are extracted to form a training segment. Each training segment is a 16 x 512 matrix that is used to model the temporal features of the respective flame candidate. We extracted more than 10,000 fire flame segments and nonfire moving objects from the video dataset to train the LSTM network. These segments are divided into three parts: 60% are used for training, 20% are used for validation and 20% are used for testing. We used the cross-entropy loss function during training, which was optimized by the stochastic gradient descent (SGD) algorithm. The learning rate was adjusted after each epoch to better learn the model parameters.

IV. EXPERIMENTAL RESULTS
ResNet-18 is used as our backbone network for extracting spatial features. The ResNet-18 architecture can balance the accuracy and network complexity. We implemented an experiment to prove the advantage of the network in Table 1.

ResNet-18 can achieve an accuracy of 93.8% with 11.4M parameters while VGG-19 can achieve an accuracy of 96.2%; however, the complexity of VGG-19 is much higher than that of ResNet-18. The accuracy of these popular models is calculated using our collected dataset of 60,000 images (30000 fire videos and 30000 nonfire videos).

By performing experiments with different network architectures of LSTM variants, as shown in Table 2, we found that BiLSTM has a better result than a single LSTM model. With more than two layers, the performance did not considerably improve while the processing time increased. Therefore, BiLSTM is the optimal architecture for our system. In addition, the proposed method, which has 14003394 parameters, can achieve an inference time of 25 ms with the setup of an Intel core i9 CPU, Windows 10, and an NVIDIA GeForce RTX2070 Super GPU.

We tested our algorithm on a large dataset with different scenarios, including large fires, medium fires, small fires, close fires, distant fires and nonfire moving objects. We have also compared our method with previous methods on our collected dataset, which includes 1032 videos (534 videos have fire, and 498 videos have no fire). Table 3 shows the detailed descriptions for a part of the full dataset. Figure 11 shows the real fire frames in the videos of the testing dataset, which demonstrates the diversity of the data. Fires appear in various shapes and environmental conditions. In the first row, the fires are large and occupy most of the area of the image frames. However, the fires are small and only a few pixels in the third row images. Unlike Figure 11, Figure 12 shows fire-like videos, which include fire-like objects such
TABLE 2. LSTM variants comparison.

| Model               | Number of parameters | Accuracy (%) | Inference time (ms) |
|---------------------|----------------------|--------------|---------------------|
| Resnet-18 + 1 layer-Bi-LSTM | 12426434             | 86.7         | 20                  |
| Resnet-18 + 2 layer-Bi-LSTM | 14003394             | 92.0         | 25                  |
| Resnet-18 + 3 layer-Bi-LSTM | 15580354             | 92.7         | 31                  |
| Resnet-18 + 4 layer-Bi-LSTM | 17157314             | 93.3         | 36                  |

TABLE 3. Test dataset description.

| Video name | Resolution | Total time | Description | Level |
|------------|------------|------------|-------------|-------|
| Test01     | 320x240    | 2 minutes  | Fire in the kitchen | Large fire |
| Test02     | 427x240    | 20 seconds | Fire accident | Large fire |
| Test03     | 400x256    | 16 seconds | Forest fire | Large fire |
| Test04     | 400x256    | 16 seconds | Forest fire | Large fire |
| Test05     | 640x480    | 3 minutes  | Out door fire | Medium fire |
| Test06     | 640x480    | 2 minutes  | Out door fire | Medium fire |
| Test07     | 320x240    | 2 minutes  | Out door fire | Medium fire |
| Test08     | 320x240    | 3 minutes  | Out door fire | Medium fire |
| Test09     | 320x240    | 2 minutes  | Out door fire | Medium fire |
| Test10     | 1280x720   | 3 minutes  | Out door fire | Medium fire |
| Test11     | 1280x720   | 3 minutes  | Out door fire | Small fire |
| Test12     | 1280x720   | 5 minutes  | Out door fire | Small fire |
| Test13     | 1280x720   | 3 minutes  | Out door fire | Small fire |
| Test14     | 320x240    | 3 minutes  | Out door fire | Small fire |
| Test15     | 1280x720   | 2 minutes  | Out door fire | Small fire |
| Test16     | 320x240    | 3 minutes  | Moving car with tailing flashing | No fire |
| Test17     | 320x240    | 3 minutes  | Orange human walking | No fire |
| Test18     | 320x240    | 2 minutes  | Orange human walking | No fire |
| Test19     | 320x240    | 3 minutes  | Orange human swaying around | No fire |
| Test20     | 1280x720   | 5 minutes  | Warning light | No fire |
| Test21     | 1280x720   | 5 minutes  | Warning light | No fire |
| Test22     | 1280x720   | 5 minutes  | Yellow human walking around | No fire |
| Test23     | 1280x720   | 5 minutes  | Warning light | No fire |

FIGURE 11. Fire video examples in testing dataset.

as moving cars and moving human and red colored objects. These samples are used to test the ability of the algorithm to eliminate false fire detection. In the Table 4, [1], [12], [20] are weak at eliminating false fire detections, resulting in degraded performance of the overall system.

For evaluation, we also implemented previous algorithms and then conducted the test using the same dataset for comparison. The algorithms that were selected included the methods of [1], [12], [19], [20], [29]. In the case of large and medium fires, the visual features of fire flames, such as color or brightness, are outstanding, so the detection rate for most algorithms is very good. All tested algorithms worked well for large and medium fires with a detection rate of 100%; however, for small fires or distant fires, where fire features are not dominant compared to other objects in the scene, some algorithms do not perform well. False detection is also a problem in fire detection. In fact, there are many moving objects with similar characteristics to fire flames. Especially in some cases (test20), even human eyes cannot recognize whether an object is a fire in a single image; therefore, the algorithms that rely only on spatial features for detecting fires will have many false detections and missed real fires. The following paragraphs provide more details about our experiments. The author of [1] detected fire flames based on
color and movement attributes. Then, the temporal variation in the fire intensity and the spatial color variation are analyzed to eliminate false detection. These algorithms use heuristic rules and thresholds to make decisions. It is not easy to select a suitable threshold. If we apply a lenient threshold to increase the detection rate, the false alarm rate could increase; conversely, if we use a strict threshold to prevent false detections, many true fires may be missed. Furthermore, these algorithms ignore spatial structures that identify object shapes, so they sometimes cannot distinguish fires from non-fire moving objects by using only color and simple rule-based temporal analysis. Experimental results show that these algorithms have good detection rates but also have many false detections. Similarly, [12] uses color and temporal variation to detect fires. The difference between the methods of [12] and [1] is that the author of [12] uses a 1-D temporal wavelet transform and a 2-D spatial wavelet transform to eliminate false detections. Our experiment shows that these algorithms also have the same problems as the method of [1], and there are many false detections.

Ref. [19] used a CNN model to detect the occurrence of fire in videos. However, the CNN model can only characterize the spatial information of fire regions and lacks temporal features to detect fire correctly. The author of [20] first detects candidate fire regions by using color and flickering features. Cascade classification was also constructed to combine many weak classifiers for robust fire/nonfire classification. Deep learning image classification using convolutional neural networks (CNNs) was also applied to increase the accuracy of the algorithm. The algorithm produces good results regarding both detection and eliminating false detections; however, the algorithm still has the same problem as the others during test20 when the human eye cannot distinguish whether there is fire using a single image. In this case, we need to monitor an object for a period of time to make the final decision, and CNNs can only work with a single image at a time.

| Video | XiaoJun Qi [1] | Toyerin [12] | NVHuan [29] | DungNM [20] | Sharma [19] | Our proposed method |
|-------|----------------|-------------|-------------|-------------|-------------|---------------------|
| Test01| Fire detected  | Fire detected| Fire detected| Fire detected| Fire detected| Fire detected       |
| Test02| Fire detected  | Fire detected| Fire detected| Fire detected| Fire detected| Fire detected       |
| Test03| Fire detected  | Fire detected| Fire detected| Fire detected| Fire detected| Fire detected       |
| Test04| Fire detected  | Fire detected| Fire detected| Fire detected| Fire detected| Fire detected       |
| Test05| Fire detected  | Fire detected| Fire detected| Fire detected| Fire detected| Fire detected       |
| Test06| Fire detected  | Fire detected| Fire detected| Fire detected| Fire detected| Fire detected       |
| Test07| Fire detected  | Fire detected| Fire detected| Fire detected| Fire detected| Fire detected       |
| Test08| Fire detected  | Fire detected| Fire detected| Fire detected| Fire detected| Fire detected       |
| Test09| Fire detected  | Fire detected| Fire detected| Fire detected| Fire detected| Fire detected       |
| Test10| Fire detected  | Fire detected| Fire detected| Fire detected| Fire detected| Fire detected       |
| Test11| Fire detected  | Fire detected| Fire detected| Fire detected| Fire detected| Fire detected       |
| Test12| Fire detected  | Fire detected| Fire detected| Fire detected| Fire detected| Fire detected       |
| Test13| Fire detected  | Fire detected| Fire detected| Fire detected| Fire detected| Fire detected       |
| Test14| Fire detected  | Fire detected| Fire detected| Fire detected| Fire detected| Fire detected       |
| Test15| Fire detected  | Fire detected| Fire detected| Fire detected| Fire detected| Fire detected       |
| Test16| False detected | False detected| No fire      | No fire      | No fire      | False detected     |
| Test17| False detected | False detected| No fire      | No fire      | No fire      | False detected     |
| Test18| False detected | False detected| No fire      | No fire      | No fire      | False detected     |
| Test19| False detected | False detected| No fire      | No fire      | No fire      | False detected     |
| Test20| False detected | False detected| No fire      | False detected| False detected| False detected     |
| Test21| False detected | False detected| No fire      | False detected| False detected| False detected     |
| Test22| No fire        | No fire      | No fire      | False detected| False detected| False detected     |
| Test23| No fire        | No fire      | No fire      | No fire      | False detected | False detected     |
TABLE 5. Quantitative comparison of different methods.

| Method         | Precision | Recall | F1 Score |
|----------------|-----------|--------|----------|
| XiaoJun Qi [1] | 0.780     | 0.830  | 0.794    |
| Turein [12]    | 0.780     | 0.800  | 0.789    |
| NVHuan [29]    | 0.870     | 0.840  | 0.854    |
| DungNM [20]    | 0.890     | 0.850  | 0.867    |
| Sharma [19]    | 0.860     | 0.810  | 0.843    |
| Proposed method| 0.920     | 0.910  | 0.915    |

To solve the problem that the CNN model has in [20], the author of [29] applied the CNN-LSTM combination model to classify sequences of images. In these algorithms, the sequences of images are directly classified into fire or nonfire categories. However, these algorithms use an entire image to train the CNN-LSTM model, which leads to problems when detecting small fires in which the visual features of fires do not dominate in scenes. This leads to the algorithm sometimes modeling the dynamic textures of moving objects rather than modeling the movement of the fire flames in the image sequences. Experimental results show that these algorithms missed many real but small fires.

By applying multistage fire detection, our proposed algorithm achieved the best results for small fires and for avoiding the false detection of nonfire moving objects. Table 4 demonstrates the qualitative results of our method and different methods evaluated with a small part of our full dataset (23 videos).

In addition to the qualitative evaluation of different methods, we conduct a quantitative evaluation of the methods on the full dataset. The evaluation metrics used are the precision, recall, and F1 score via 10-fold cross-validation. The detailed results are given in Table 5. Our proposed method can achieve the highest F1-score, which proves that our proposed method can detect fire correctly with a low false-positive rate. Other methods achieve high accuracy; however, the F1-scores are low due to high false-positive rates, such as for the methods of [12], [1]. The methods of [20], [29], [19] achieve low recall because these methods are not suitable to detect the small fire objects in videos, such as those that occur frequently in our dataset.

In the performance test, we measure the processing time for every step of the algorithms. The time needed for visual CNN feature extraction is approximately 10 milliseconds, and the time to run LSTM sequence image classification is approximately 10 milliseconds, which is fully suited to real-time applications.

V. CONCLUSION AND FUTURE WORK

Automatic fire detection is very important as prompt fire warnings give people a better chance to escape and reduce damage from fires. This paper introduces a video-based fire detection algorithm for use in an early fire-alarm system. This algorithm uses a combination of a CNN and an LSTM network to analyze fire flames in both the spatial and temporal domains. By exploiting the advantages of both CNNs and LSTM in computer vision tasks, our algorithm achieved a significant improvement over the existing methods. The experimental results show that our algorithm is fast, reliable and fully suited for use in real-time surveillance systems. Furthermore, the advantages of our proposed method can be highlighted as follows:

- The proposed model is threshold-free and therefore suitable for application in various weather conditions.
- The proposed model can achieve high accuracy with a low false alarm rate.
- Unlike other methods that apply fire classifiers to entire images, resulting in missed detections of small fires, our proposed method incorporates a fire candidate extraction stage so that our system can detect various sized fires. In addition, our method can be fast because of processing small cropped fire images as the input of the CNN-LSTM model.
- We collected the fire video dataset from two main sources: crawling fire videos from the internet and fire videos from real-world deployment of our proposed method in various weather conditions. Our dataset will be made available for research purposes.

Despite the positive results, our proposed method still has several limitations such as detection of fire in different colors (blue fire, white fire), unstable basic features for extracting fire candidates, and lack of fire image data. In future work, we will improve the proposed method as follows:

- We will collect more data during the deployment of the proposed system in the various real-world environments, especially for blue and white fires.
- We will integrate a fire candidate segmentation model to eliminate the instability of current basic features.
- We will investigate the end-to-end model training scheme by changing the annotation of fire flames.
- We will investigate how to integrate smoke detection to improve the early fire detection ability.

ACKNOWLEDGMENT

This simulation was implemented by using GPU workstation (DGX Station, NVIDIA) and Carsim at the Future Automotive Intelligent Electronics Core Technology Center.

REFERENCES

[1] X. Qi and J. Ebert, “A computer vision based method for fire detection in color videos,” Int. J. Imag., vol. 2, no. S09, pp. 22–34, 2009.
[2] P.-H. Huang, J.-Y. Su, Z.-M. Lu, and J.-S. Pan, “A fire-alarming method based on video processing,” in Proc. Int. Conf. Intell. Inf. Hiding Multimedia, Dec. 2006, pp. 359–364.
[3] J. Chen, Y. He, and J. Wang, “Multi-feature fusion based video flame detection,” Building Environ., vol. 45, no. 5, pp. 1113–1122, May 2010.
[4] Z. Teng, J.-H. Kim, and D.-J. Kang, “Fire detection based on hidden Markov models,” Int. J. Control, Autom. Syst., vol. 8, no. 4, pp. 822–830, Aug. 2010.
[5] B. C. Ko, K.-H. Cheong, and J.-Y. Nam, “Fire detection based on vision sensor and support vector machines,” Fire Saf. J., vol. 44, no. 3, pp. 322–329, Apr. 2009.
[6] B. U. Turein, “Fire detection algorithms using multimodal signal and image analysis,” Ph.D. dissertation, Dept. Elect. Electron. Eng., Bilkent Univ., Ankara, Turkey, 2009.
[7] A. E. Gunawardena, R. M. M. Ruwanthika, and A. G. B. P. Jayasekara, “Computer vision based fire alarming system,” in Proc. Moratuwa Eng. Res. Conf. (MERCon), Apr. 2016, pp. 325–330.
[8] I. Chakraborty and T. K. Paul, “A hybrid clustering algorithm for fire detection in video and analysis with color based thresholding method,” in Proc. Int. Conf. Adv. Comput. Eng., Jan. 2010, pp. 277–280.
[9] T. Celik and H. Demirel, “Fire detection in video sequences using a generic color model,” *Fire Saf. J.*, vol. 44, no. 2, pp. 147–158, Feb. 2009.

[10] G. Marbach, M. Loepe, and T. Brubacher, “An imaging processing technique for fire detection in video images,” *Fire Saf. J.*, vol. 41, no. 4, pp. 285–289, Jun. 2006.

[11] T. Celik, “Fast and efficient method for fire detection using image processing,” *ETRI J.*, vol. 32, no. 6, pp. 881–890, Dec. 2010.

[12] B. U. Töreyin, Y. Dedeoğlu, U. Güdükbay, and A. E. Çetin, “Computer vision based method for real-time fire and flame detection,” *Pattern Recognit. Lett.*, vol. 27, no. 1, pp. 49–58, Jan. 2006.

[13] B. Ko, K.-H. Cheong, and J.-Y. Nam, “Early fire detection algorithm based on irregular patterns of flames and hierarchical Bayesian networks,” *Fire Saf. J.*, vol. 45, no. 4, pp. 262–270, Jun. 2010.

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 25. Stateline, NV, USA, Dec. 2012, pp. 1097–1105.

[15] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[16] A. K. Reyes, J. C. Caicedo, and J. E. Camargo, “Fine-tuning deep convolutional networks for plant recognition,” *CLEF Working Notes*, vol. 1391, pp. 467–475, Sep. 2015.

[17] M. Ogquab, L. Bottou, I. Laptev, and J. Sivic, “Learning and transferring mid-level image representations using convolutional neural networks,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 1717–1724.

[18] Y. Bengio, “Deep learning of representations for unsupervised and transfer learning,” in *Proc. ICML Workshop Unsupervised Transf. Learn.*, 2012, pp. 17–36.

[19] J. Sharma, O.-C. Grammo, M. Goodwin, and J. T. Fidje, “Deep convolutional neural networks for fire detection in images,” in *Proc. Int. Conf. Eng. Appl. Neural Netw. Cham, Switzerland*: Springer, 2017, pp. 183–193.

[20] N. M. Dung and S. Ro, “Algorithm for fire detection using a camera surveillance system,” in *Proc. Int. Conf. Image Graph. Process. (ICIGP)*, 2018, pp. 38–42.

[21] J. Sharma, O.-C. Grammo, and M. Goodwin, “Deep CNN-ELM hybrid models for fire detection in images,” in *Proc. Int. Conf. Artif. Neural Netw. Cham, Switzerland*: Springer, 2018, pp. 245–259.

[22] P. Li and W. Zhao, “Image fire detection algorithms based on convolutional neural networks,” *Case Stud. Thermal Eng.*, vol. 19, Jun. 2020, Art. no. 100625.

[23] D. Jin, S. Li, and H. Kim, “Robust fire detection using logistic regression and randomness testing for real-time video surveillance.” in *Proc. IEEE 10th Conf. Ind. Electron. Appl. (ICIEA)*, Jun. 2015, pp. 608–613.

[24] A. Ullah, J. Ahmad, K. Muhammad, M. Sajjad, and S. W. Baik, “Action recognition in video sequences using deep bi-directional LSTM with CNN features,” *IEEE Access*, vol. 6, pp. 1155–1166, 2017.

[25] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, T. Darrell, and K. Saenko, “Long-term recurrent convolutional networks for visual recognition and description,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 2625–2634.

[26] M. Z. Islam, M. M. Islam, and A. Asraf, “A combined deep CNN-LSTM network for the detection of novel coronavirus (COVID-19) using X-ray images,” *Informat. Med. Unlocked*, vol. 20, Jan. 2020, Art. no. 100412.

[27] M. S. Al-Rakhami, M. M. Islam, M. Z. Islam, A. Asraf, A. H. Sodhro, and W. Ding, “Diagnosis of COVID-19 from X-rays using combined CNN-RNN architecture with transfer learning,” *MedRxiv*, Aug. 2020.

[28] T. Akilan, Q. J. Wu, A. Safaei, J. Huo, and Y. Yang, “A 3D CNN-LSTM-based image-to-image foreground segmentation,” *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 3, pp. 959–971, Mar. 2020.

[29] X. T. Pham, H. V. Nguyen, and C. N. Le, “Real-time long short-term glance-based fire detection using CNN-LSTM neural network,” *Int. J. Intell. Inf. Database Syst.*, vol. 1, no. 1, p. 1, 2021.

[30] C. Hu, P. Tang, W. Jin, Z. He, and W. Li, “Real-time fire detection based on deep convolutional long-recurrent networks and optical flow method,” in *Proc. 37th Chin. Control Conf. (CCC)*, Jul. 2018, pp. 9061–9066.

[31] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei, “ImageNet: A large-scale hierarchical image database,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.