Evolving Systems
https://doi.org/10.1007/s12530-024-09623-3

ORIGINAL PAPER

A one stream three-dimensional convolutional neural network for fire recognition based on spatio-temporal fire analysis

Zeineb Daoud[1] · **Amal Ben Hamida**[1] · **Chokri Ben Amar**[2] · **Serge Miguet**[3]

Received: 26 January 2024 / Accepted: 1 September 2024

Abstract
Fires are among the most frequent disasters, causing serious injuries and extensive property destruction. In order to prevent the uncontrolled spread of fires, recognizing fires accurately and at an early stage is crucial, especially in video surveillance applications. The majority of the available deep fire detection models currently operate on single images, limiting their analysis to spatial features only. The temporal context and motion information, present in consecutive frames of a scene, are not involved leading to incorrect predictions throughout the video. To address this shortcoming, it is proposed in this work to explore the temporal information using deep learning networks to directly recognize fire. Indeed, a novel three-dimensional convolutional neural network, named 3D Fire Classification Network, is introduced. This approach exploits spatio-temporal features to analyze and recognize a video sequence as either fire or non-fire. Initially, the input data is processed to enlarge and diversify the constructed dataset. Then, it is passed through the designed network for training. The derived model comprises a relatively smaller number of layers, with a reduced number of parameters. The conducted experiments demonstrate the efficiency of the resulting model on the created dataset, achieving an improved accuracy of 99.23%. Furthermore, the findings show that the developed model consistently outperforms the related methods in recognizing fire videos.

Keywords Fire recognition · Three-dimensional convolutional neural network (3D CNN) · Spatio-temporal learning · Video analysis

1 Introduction

In the last few decades, the number of fires has been increased in the world, threatening the planet and people's safety causing hazardous effects and huge damages. In fact, millions of acres are burned destroying animals, trees, homes, and people. According to the Center for Disaster Philanthropy, the 43 member countries including some states in the Middle East and North Africa, where forest fires were previously infrequent, are now seeing a significant increase in the fires number and the burnt area (Center Philanthropy 2022). In 2021, more than 550,000 hectares were burned in the European Union and its Mediterranean neighboring countries including Turkey, Algeria and Tunisia. In Turkey, wildfires were around 2793 burning about 139,503 ha. In Tunisia, meanwhile, 28,493 ha were affected (Statista 2022). As reported in the Technical Report of the Mediterranean wildfires, the most affected countries by wildfires, in 2021, were Greece, Italy, followed by Portugal, Spain and France where hundreds of people were killed and over than 620,000 ha were burned in July and August (Eberle and

✉ Zeineb Daoud
zeineb.daoud@enis.tn

Amal Ben Hamida
benhamida.amal@gmail.com

Chokri Ben Amar
chokri.benamar@tu.edu.sa

Serge Miguet
serge.miguet@univ-lyon2.fr

[1] REGIM-Lab.: REsearch Groups in Intelligent Machines, University of Sfax, National Engineering School of Sfax (ENIS), BP 1173, 3038 Sfax, Tunisia

[2] Department of Computer Engineering, College of Computers and Information Technology, Taif University, P.O. Box 11099, 21944 Taif, Saudi Arabia

[3] Univ Lyon, Univ Lyon 2, CNRS, INSA Lyon, UCBL, Centrale Lyon, LIRIS, UMR5205, 69676 Bron, France

Published online: 10 September 2024

Higuera Roa 2022). An automatic detection of fire events is thereby essential for security applications and intelligent surveillance. This is particularly when considering the ineffectiveness of the available sensor-based fire detection systems (Avazov et al. 2022). With these captors, the alarm is only triggered when smoke or heat is close enough to the sensors. In certain situations, such as expansive open areas or high-temperature environments, using a sensor-based detection system becomes impractical as it leads to frequent false alarms (Avazov et al. 2022). In addition, these systems lack the capability to provide visual information, which may be essential for helping firefighters in quickly understanding the fire scene. Important details concerning the fire size, locations and behavior are also not provided.

To cope with all these shortcomings, researches and development efforts have been carried out to reduce false alarms risks and to achieve accurate real-time fire detection early-stage. This has become particularly significant with the advancement of video surveillance systems. Add to that, the computer vision techniques and the recent evolution of neural networks are frequently employed in such applications (Çetin et al. 2013; Muhammad et al. 2019). In fact, numerous significant studies related to fire recognition have been proposed over the years, including the video-based fire detection systems (VFD) and the deep learning-based fire detection systems. VFD approaches are the first developed works as an alternative solution to the existing sensor-based systems (Chen et al. 2004; Celik 2010). The detection process involves a manual extraction of flame features, such as color, texture, shape, and motion (Khalil et al. 2021; Wahyono, Harjoko et al. 2022). This extraction step, the features analysis, the subsequent detection and classification processes are time-consuming, which limits the use of these VFD methods in real-time. To avoid hand-crafted operations and to ensure automatic fire detection task, deep learning (DL) models have been extensively employed in recent times (Bhat and Khan 2022; Gayathiri e al. 2023; Harsha et al. 2023). They have proven their efficiency in different fields, including object recognition, machine learning (Manohar and Das 2022), parameters estimation (Manohar and Das 2023), and medical analysis (Lakshmi et al. 2024; Manohar and Das 2023). The features they extract are learned automatically from annotated data rather than hardcoded by the developper. Hence, these DL models can be adapted for an automatic fire recognition in surveillance applications, as presented in this paper.

It is therefore interesting to develop a real-time fire recognition method based primarily on exploiting both the spatial and temporal information in video sequences. It is noticed that the most developed models in the literature are designed by exploring spatial features, which provide the visual appearance and contextual information of the data. Spatial features are directly extracted from frames through two dimensional convolutional neural networks (2D CNNs). However, these models lack temporal features, that capture the motion dynamics occurring in the range of video frames. This is since the fire event is considered as an action in video sequences, distinguished by its spatial and temporal features across successive frames. The objective of fire recognition is to effectively learn discriminative spatio-temporal representations from video sequences to identify the fire class. Motivated by these claims, the direct learning of spatio-temporal features from video frames is suggested in this work. It is achieved using three-dimensional networks (3D CNNs). With much interest to this, the novelty of this contribution is to recognize fire in surveillance videos by designing a three-dimensional network, known as 3D Fire Classification Network "3D FireClassNet". The presented approach starts by preprocessing the input data for its for enlargement and diversification. Then, this preprocessed data is fed through the novel 3D network for training. The designed architecture has the capability to be directly applied to consecutive frames, for the extraction and learning of spatio-temporal features.

The remainder of this paper is organized as follows. Section 2 provides a background on the existing deep learning architectures, used for spatio-temporal analysis. In Sect. 3, a literature review of DL-based approaches designed to recognize fire in video sequences is presented. Section 4 thoroughly describes the details of our proposed method with the novel spatio-temporal convolutional neural network. The experimental findings and discussions are exhibited in Sect. 5, including a comparative study with the state-of-the-art works. Lastly, Sect. 6 deals with the conclusions of this paper.

2 Deep learning architectures for spatio-temporal analysis

In general, videos are constructed from spatial and temporal domains, providing much more information content, compared to a single image. Spatio-temporal features extraction methods can be categorized as either hand-crafted or automated (Rasool Abdali and Ghani 2019; Mehta and Singh 2023). Deep learning networks offer the ability to automatically capture these features, producing promising results in various spatio-temporal approaches for different applications. The following subsections outline the most common architectures employed in deep learning for capturing and learning spatio-temporal features.

2.1 Two-stream architectures

In order to exploit both spatial and temporal features, several researchers have proposed the development of two-stream

architectures. These architectures consist of two separate CNNs (Simonyan and Zisserman 2014). Each one serves for a specific purpose. The first convolutional neural network is the spatial stream, dedicated to handle spatial features. The second one is the temporal stream, devoted to handle temporal features. The outcomes of these distinct networks can be merged to create a spatio-temporal video representation.

As shown in Fig. 1, the spatial stream takes a single frame from the video to pass it through a series of CNN kernels. Then, using the extracted spatial information, predictions are generated. The second temporal stream operates by gathering optical flows of each frame. Using this motion data, predictions are produced by the final fully connected and SoftMax layers of the temporal network. Each individual stream is constructed using a deep ConvNet. The softmax scores produced by these networks are merged by the use of the late fusion technique. With this technique, the two streams are independently trained and combined just before the model makes the last decision. The final probability is attained by averaging the predicted probabilities derived from both streams.

Relying on an external optical flow algorithm is one of the disadvantages of this type of architectures. This algorithm needs to be executed to compute the motion vectors for each video, it is performed before the training phase. Being coupled, coupled with the training processes for the two networks, leads to a substantial amount of time required for the design of the final model.

2.2 Convolutional neural network and long short-term memory (CNN-LSTM)

A second type of deep learning architectures is the association of a Long Short-Term Mermory (LSTM) to a CNN for video analysis. LSTM is a variant of recurrent neural networks (RNNs), introduced by Hochreiter and Schmidhuber (1997). It is designed to enhance its ability to retain information for long periods. It is structured as a sequence of recurrent cells linked together, where each cell is connected to the next one through a cell state (C). This property makes LSTM well adapted to tasks requiring long-term memorization (Karthika et al. 2023). That's why some researchers propose the training of LSTM networks to create temporal models.

The combination of an LSTM with a CNN enables the extraction and learning of spatio-temporal features from videos. This architecture is illustrated in Fig. 2. The CNN acts as an extractor of spatial features. Indeed, the individual images in a video sequence are fed into a CNN model, which in turn extracts spatial features. These features are subsequently passed through the LSTM layer. The output derived from the LSTM layer is connected to a fully connected layer, resulting in the recognition purpose. The main goal of incorporating LSTM is to capture the temporal connections among images by retaining a memory of preceding frames. This enables the model to understand and use the sequential information presented in the video data.

This architecture may take advantage of transfer learning by exploiting a pre-trained CNN model, such as VGG-16, VGG-19, ResNet, and others to extract spatial features. The transfer learning is an effective method for building accurate models, particularly in cases of limited data. As a result, a combination of CNN and LSTM shows its robustness in learning spatio-temporal features and developing efficient models for video analysis. Nevertheless, it's important to acknowledge that a CNN needs significantly long training time to fine-tune the vast number of model's parameters. This issue becomes more complex when considering the extension of temporal aspects in the architecture. This is because the network requires to process not just single frames, but also several video frames simultaneously. Similarly, LSTM training takes a large time, since it has more parameters (Kanna and Santhi 2021). Add to that, it needs more memory requirements.

2.3 Convolutional long short-term memory (ConvLSTM)

Another category of deep learning architectures designed for video recognition is the convolutional long short-term

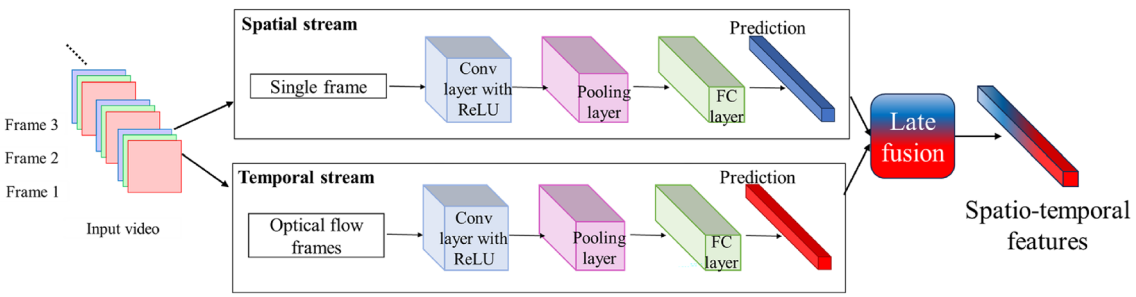

Fig. 1 Overview of the two-stream architecture for video recognition

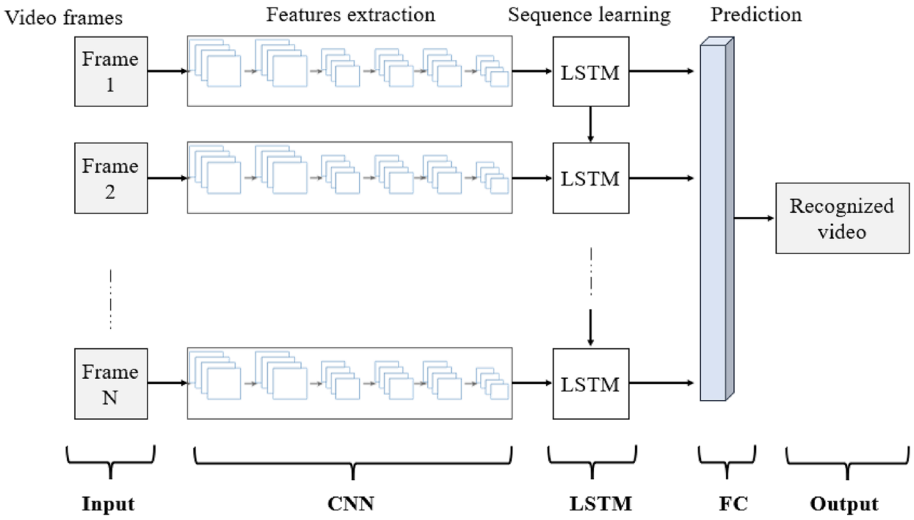

Fig. 2 Overview of the CNN-LSTM architecture for video recognition

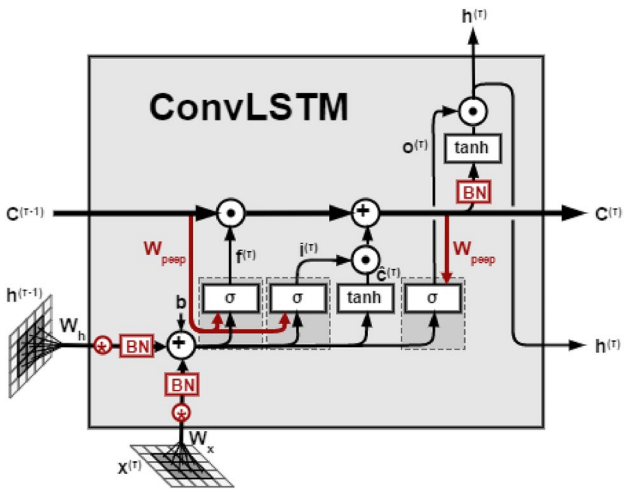

Fig. 3 A ConvLSTM cell (Verlekar and Bernardino 2020)

memory (ConvLSTM). It represents a variant of the LSTM architecture that integrates convolutional operations within the structure of the LSTM cell (Shi et al. 2015). These convolutional operations are introduced during transitions between layers. They replace the internal matrix multiplications of the LSTM (Verlekar and Bernardino 2020), as shown with the red color in Fig. 3. Thus, the information passing through ConvLSTM cells retains the input dimension, enabling the network to achieve better spatio-temporal correlations (Verlekar and Bernardino 2020; Kanna and Santhi 2022). The ConvLSTM architecture for video recognition is depicted in Fig. 4.

It should be noted that both ConvLSTM and CNN-LSTM architectures serve the same functional purpose: extracting spatio-temporal features from video data for video recognition tasks. However, they differ in structure. Indeed, ConvLSTM incorporates convolution in the architecture, whereas CNN-LSTM externally combines the two types of networks by concatenating their outputs together. As defined, the ConvLSTM network effectively captures localized spatio-temporal correlations. It accomplishes this by employing a convolution operator to predict the future state of a specific cell in the grid using inputs and previous states of its local neighbors (Vrskova et al. 2022).

ConvLSTM proves to be suitable for processing images and videos with temporal dependencies, achieving significant results. But, its weaknesses lie in the substantial computational demands and high memory consumption.

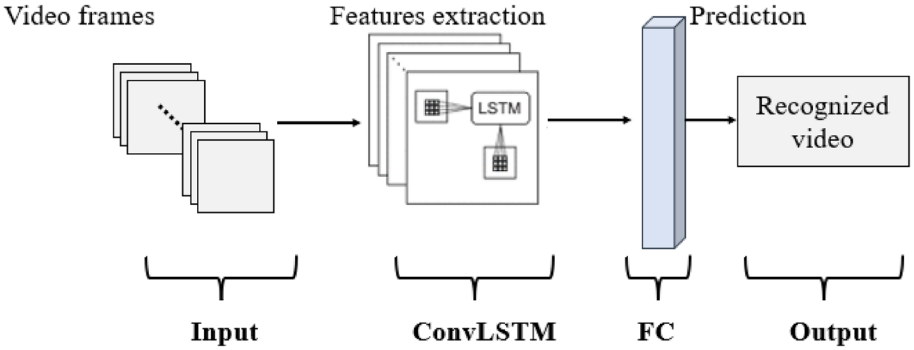

Fig. 4 ConvLSTM architecture for video recognition

2.4 Three-dimensional convolutional neural network (3D CNN)

Among the deep learning architectures used for video recognition, there are the 3D convolutional neural networks (3D CNNs), known as a spatio-temporal networks. They play a prominent role in exploiting spatial and temporal features (Tran et al. 2015). The idea involves expanding the 2D spatial CNN, based on Conv and pooling layers, into a 3D spatio-temporal CNN. This extension aims to analyze and then recognize videos.

3D CNNs are similar to 2D CNNs, but with two primary differences. First, they are designed to capture the temporal relations between video frames, by using three-dimensional kernels. This is achieved by processing sequences of frames rather than individual ones. Additionally, a 3D CNNs can learn the three-dimensional features of video sequences, and generate 3D feature maps, a capability that is impossible with 2D CNNs. An example of a 3D CNN architecture is presented in Fig. 5. As it is displayed, a 3D CNN can be composed by a succession of Conv3D layers, ReLU activation functions, and 3D pooling layers.

- **3D convolution layer (Conv3D)** In the same way that the convolution layer (CONV or Conv2D) is the basic component of the CNN, the three-dimensional convolution layer (Conv3D) is equally the fundamental element of the 3D CNN. Indeed, the CONV layer lacks the temporal information in each convolution operation. In contrast, the 3D convolution maintains the temporal information from the input data, producing 3D feature maps as an output volume. The input to Conv3D is convolved across four dimensions: two spatial dimensions (width and height), one channel dimension, and one time dimension (frame). During the convolution process, the 3D CNN generates a three-dimensional activation map. This feature map serves for data analysis and for the incorporation of temporal context. In this operation, three-dimensional filters are applied, where the kernel moves along three directions, as visually depicted in Fig. 6b. The resultant output has the form of a 3D volume space.

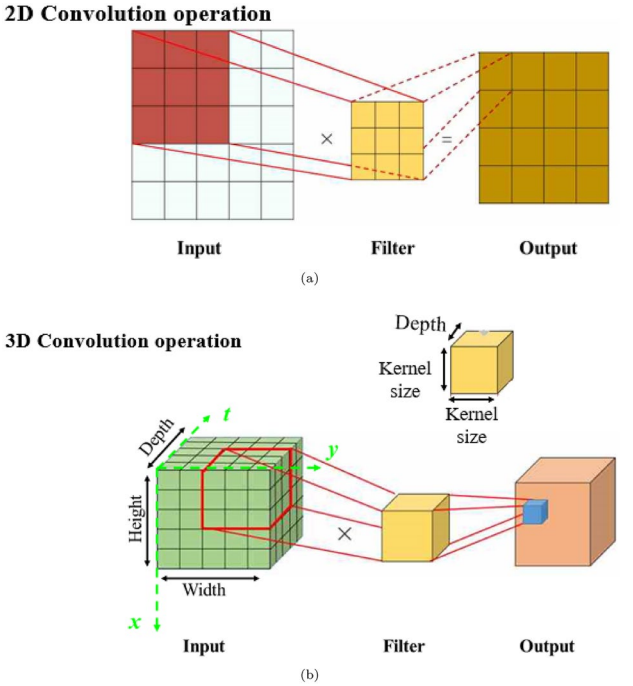

Fig. 6 The difference between 2D and 3D convolution operations: **a** 2D convolution operation, **b** 3D convolution operation

The 3D convolution operation is accomplished by wrapping around the center of a cube and stacking adjacent layers on top of each other (Vrskova et al. 2022). The motion information is captured by the interconnections between the feature maps.

- **3D Pooling layer** The 3D pooling layer has the same purpose as the pooling layer used in 2D CNN structure. It acts as a nonlinear down sampling operation for an input tensor. Its aims to reduce the spatial dimensions of an image, while retaining only the most significant pixels. When applying a 3D pooling in the 3D neural network construction, the pooling size must consist of three values, reflecting the 3D data being dealt with. In fact, this operation involves the division of the input tensor's data into smaller 3D subtensors along all three dimensions. Afterwards, in each subtensor, the element with the high-

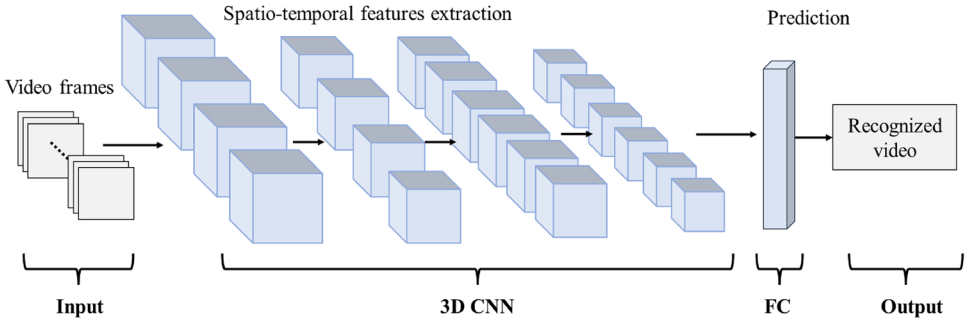

Fig. 5 3D CNN architecture for video recognition

est numeric value is selected. This process converts the input tensor into an output tensor, in which each sub-tensor is replaced by its respective element (maximum, minimum or average element). This is because there are three commonly employed techniques: Max pooling (selecting the highest value), Min pooling (selecting the lowest value), and Average pooling (calculating the average of the values). MaxPooling3D is frequently used in the case of color images and its visual representation can be seen in Fig. 7.

This exploitation of temporal context is a notable advantage of 3D CNNs in video analysis. This is thanks to the four dimensions (two spatial dimensions, one channel dimension and one temporal dimension), that allow all types of temporal interactions between adjacent frames to be easily learned. The 3D CNN architecture is not only uncomplicated, but it is also fast, and easier to train, particularly when compared to CNN-LSTM. Especially with sufficient data, the 3D CNN is the most efficient architecture, as a spatio-temporal network for video recognition. A limitation of this 3D CNN is that the increase of the input dimensions leads to a significant rise in both memory and computational requirements (Tran et al. 2015).

3 Related works

In the literature, the aforementioned deep learning architectures, from the previous section, have been suggested for recognizing fires in videos, through spatio-temporal analysis. Hence, the DL-based fire recognition approaches have been considered as a significant challenge because of specific information nature contained in videos, particularly the temporal continuity of fire movement. Indeed, it is not only about the basic two-dimensional space of a frame; but the incorporation of previous and subsequent frames is also crucial to effectively capture the temporal information.

The spatio-temporal two-stream convolutional neural network is employed in Shin et al. (2018) to introduce a fire recognition model. A spatio-temporal two-stream convolutional neural network-based fire recognition method is introduced.

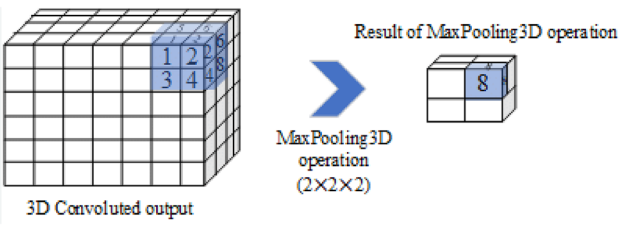

Fig. 7 MaxPooling3D operation (Vrskova et al. 2022)

The spatial stream employs the VGG-16 network, while the temporal model is built using the 3D CNN network. For both streams, transfer learning is applied. The output vectors from each stream are concatenated through the fusion method in the fully connected layer. With this type of architecture, the accuracy of the obtained classification model is enhanced, and the false positives are reduced. The proposed model demonstrates superior performance compared to the 2D CNN model. However, its computational cost is high. Another approach, presented in Rasool Abdali and Ghani (2019), is based on the CNN-LSTM architecture for a real-time fire detection. Initially, a CNN is applied to spatially extract features. Then followed by the LSTM cells, the temporal relation learning method is used to build the model. To feed the data, the time distribution technique is utilized. Experimental results show an improvement in fire recognition performance, achieving a promising level of accuracy.

To analyze and recognize the flame regions in both spatial and temporal domains, this CNN-LSTM architecture is also used in Abhilash (2023). Herein, a fire candidate extraction stage is introduced to detect fires of varying sizes. Then, the CNN-LSTM is employed to analyze the small and clipped fire images. Despite the achieved effectiveness, this approach suffers from the instability of the fundamental features (Abhilash 2023).

In the same way, the spatial and temporal features are exploited by Zhikai Yang et al. in Yang et al. (2020). Three novel fire recognition models are developed, based on two lightweight CNNs (ResNet-18 and MobileNets) and the Simple Recurrent Unit (SRU). The SRU, which is improved from LSTM by adding a reset gate, is a variant of Recurrent Neural Network (RNN). The first model combines ResNet-18 with the SRU, and MobileNet is merged with SRU for the built of the second model. In the third one, a 3D Conv layer is added between the MobileNet and SRU components, taking into account the indoor settings and flame characteristics. The resulting models prove their efficiency, compared to the other CNN-based fire recognition models, via single frame. It is also demonstrated that the third developed model (MobileNet + 3D Conv layer + SRU) reaches the best performance. Hence, the importance of the temporal aspect is confirmed.

Likewise, another type of DL architecture, using a CNN combined with a ConvLSTM, is presented in Verlekar and Bernardino (2020) to classify video sequences into either fire or non-fire categories. A frame selection step is firstly applied in this approach to process video sequences of variable durations. Indeed, 15 frames are selected from every input scene. Each frame is then fed into a CNN, named Xception, in order to extract static features. Subsequently, spatio-temporal features are extracted using a ConvLSTM. Finally, the output of the last cell of the ConvLSTM is passed to the fully connected network for the classification

purpose. According to this method, the achieved classification accuracy validates the effectiveness of the proposed model in classifying video sequences.

The use of ConvLSTM models is also shown in Masrur et al. (2024), where they identify and exploit the scale-dependent spatiotemporal interconnections between the space-time of the fire event. Two attention-based spatiotemporal models are proposed in this study for predicting the wildfire progression. Another related work (He et al. 2024) combines ConvLSTM with a CNN and a Vision Transformer (ViT) for fire recognition. In He et al. (2024), the CNN-based network is used to extract local features, while the Vision Transformer (ViT) serves as a global feature extraction method employing multi-head attention to gather information across the entire image. Combining CNN with ConvLSTM adds the ability to consider image data features within a spatiotemporal context. Promising findings are reached demonstrating the effectiveness of the presented approach. However, the model's precision should be further optimized by integrating more dependable datasets.

A spatio-temporal network, designed specifically for night-time wildfire recognition in videos, is provided in Agirman and Tasdemir (2022). The used architecture combines a CNN with a variant of LSTM, known as the bidirectional long short-term memory (BLSTM). In fact, a BLSTM cell employs two interconnected LSTM cell engines. The proposed network initially extracts spatial features from videos, through the use of a pretrained GoogLeNet network, with transfer learning applied to this CNN. These extracted features are fed into a BLSTM network for temporal learning. The final layers of the proposed architecture consist of a fully connected layer with two output classes, followed by a SoftMax layer for probability calculations. Despite the promising detection results, instances of misclassification persist.

Similarly, another DL-based fire recognition approach is proposed in Vu et al. (2021). In this method, flame regions are initially detected by exploiting the motion and color features of the fire. In the subsequent phase, a combination of a CNN and LSTM is applied to determine whether the flame is a true fire or a non-fire moving object. In fact, the CNN architecture is based on ResNet-18 to extract spatial features. The LSTM model is used to extract temporal features in videos. Experimental results demonstrate that the suggested approach performs well in terms of accuracy, with rapid processing speed. Aiming to further improve the obtained results, authors suggest in Nguyen et al. (2021)an enhanced multistage fire detection method. It is also based on the CNN-LSTM architecture. As the previously reported work (Vu et al. 2021), the candidate fire regions are firstly detected using both a color model and the computed flicker energy. The images from every candidate flame region are then passed through the convolutional network, to extract spatial features. For this purpose, the CNN, based on the pretrained ResNet-18 model, is finetuned. These features are later on input to a multilayer Bidirectional LSTM network, which temporally merges the extracted information. This BiLSTM is dedicated to classify the fire/non-fire sequence images. The outcome of the proposed method achieves the highest F1-score, validating its effectiveness in accurately recognizing fires with minimal false positives. However, the limited availability of fire image data remains a limitation of this method, which ultimately impacts its overall performance.

After the above discussion of different deep learning approaches, it is clear that the task of recognizing fire in video surveillance scenes presents a spatio-temporal challenge. It can be concluded that these works exhibit a reliable performance, when compared to deep learning approaches devoted for fire images classification. Accurately capturing the spatial and temporal features of the flame object, yields to achieve good results. Besides, it is deduced that the exploitation of the spatio-temporal features, particularly through the use of deep learning architectures, remains limited in the fire recognition filed. Nevertheless, the presented related works frequently encounter issues of high cost in terms of both time and computation. This is due to the structure of the spatio-temporal architectures, leading to a high parameters number. Consequently, the memory requirements are increased, owing to the heavy architectures and the large number of video frames, needed for the spatial and temporal training. Therefore, up to now, developing a spatio-temporal architecture with one stream network and fewer trainable parameters is still the ongoing researchers focus. This aim is to mitigate the challenges posed by memory and computational demands.

4 3D FireClassNet: a novel spatio-temporal convolutional neural network for fire recognition in video surveillance scenes

The suggested approach is a novel DL-based method for fire recognition. It is based on the exploration of spatial and temporal information, available in a fire video surveillance scenes. It is designed considering the 2D convolutional neural network "FireClassNet", presented in Daoud et al. (2023), which exhibits highly efficacy for static image recognition tasks. The effectiveness of this 2D CNN lies in its ability to process spatial information in fire images. However, fire videos inherently contain both spatial and temporal data, which 2D CNN struggles to capture. Because of this limitation, the "FireClassNet" model overlooks crucial motion information in fire videos, making it unsuitable for fire video analysis task (Daoud et al. 2023). As a result, it

may not correctly predict every video frame, due to rapid changes between successive frames.

Unlike 2D CNN structure, which focuses only on spatial information, the 3D CNN structure adds a third dimension, time, enabling it to capture spatio-temporal information in videos. The motion and changes across frames are detected through the 3D CNN making it ideal for fire video recognition task. The goal of this study is to create an end-to-end 3D CNN designed to classify videos as fire or non-fire by extracting spatial and temporal features. Our major motivation is to remedy the limitation of the "FireClassNet", the 2D CNN architecture presented in Daoud et al. (2023), and to automatically recognize fire videos while improving accuracy and decreasing false alarms number. To achieve this, it is suggested to design 3D CNN structure named "3D FireClassNet", inspired by the "FireClassNet" architecture. This structure considers the temporal dimension that enhances the capability of the network to recognize complex relationships between spatial and temporal features in videos. Nevertheless, it is important to take into account the increased computational requirements of a 3D CNN architecture. Hence, another objective of this work is to develop the 3D CNN model with fewer parameters, to be used even in mobile systems with limited memory and processing capacity.

The different phases of the proposed approach are displayed in Fig. 8. The process begins by preprocessing the created dataset to enlarge the input data. Subsequently, the development of the fire recognition model, is carried out involving a novel deep learning network that automatically extracts spatio-temporal features. With these extracted features, the model is trained and then tested on a new data. The performance of the designed model is evaluated, using different evaluation metrics. The details of each phase are described in the following two subsections.

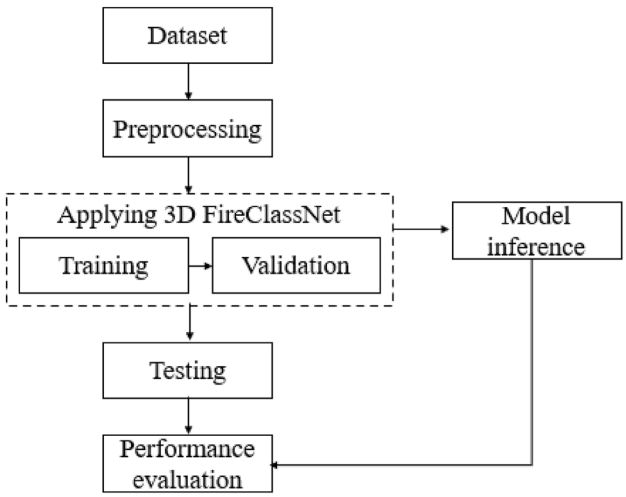

Fig. 8 Flow chart of the proposed fire recognition approach

4.1 Preprocessing

Before feeding the data to the designed network, preprocessing is conducted. This phase is crucial, as it has a significant impact on overcoming the shortcomings of the created data, such as insufficient data and its lack of diversity. The process involves making essential adjustments and transformations to the data. Since the deep learning models need large amounts of data, it is proposed in our approach to generate additional data by augmenting the collected dataset. Specifically, video data augmentation techniques, including horizontal and vertical flipping and rotation, are employed to expand our dataset and enhance its diversity. Some samples are shown in Fig. 9.

Each video of the constructed dataset undergoes various augmentations, including horizontal flips, vertical flips, and rotations. The augmented dataset, now comprising a diverse set of samples, is then used to train the deep learning model. This preprocessing phase not only enhances the data quantity by introducing different variations of each sample but also improves the model's performance by mitigating overfitting. By augmenting the dataset in this way, we ensure a richer training set for the model, which should lead to improved results and better generalization.

4.2 Presentation of the "3D FireClassNet" architecture

As it is observed in the literature review, the existing works for fire recognition in videos, particularly those that exploit the spatio-temporal features, are limited and still in progress. Most of works use the combination of two networks: the first one is trained with spatial features and the second one is trained with temporal features. This results in a high time consumption and large memory requirements. In order to explore spatial and temporal features in a single network and to avoid the huge training and prediction times, a novel deep fire recognition model is needed nowadays. Motivated by these statements, the main contribution of the presented approach is the development of a 3D CNN architecture for fire videos recognition, by exploiting spatio-temporal features.

Inspired by the 2D CNN "FireClassNet", suggested in Daoud et al. (2023), our 3D CNN, named "3D FireClassNet", is designed to simultaneously handle spatial and temporal features. The differences are evident in the convolution and pooling layers, as well as in the employed filter kernel sizes, with the addition of the temporal dimension. Hence, the kernel of a 3D convolution layer is expressed as ($H \times W \times dt$), where H and W represent the height and width of the convolution kernels on the 2D plane, and dt denotes the depth of the convolution kernel, representing the time dimension. The 2D and 3D structures of the convolutional

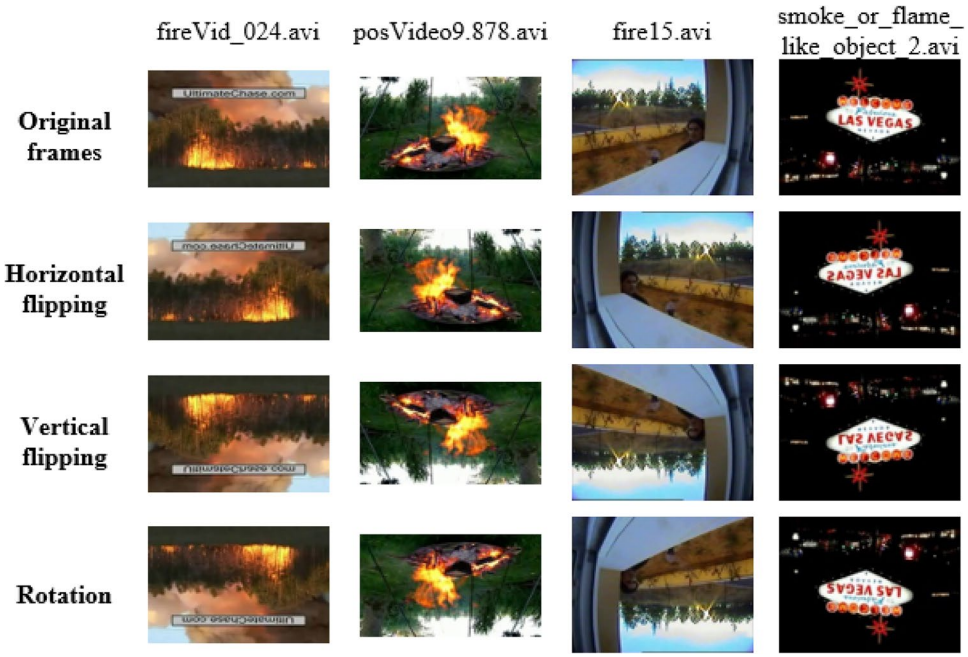

Fig. 9 Examples of the augmented data

layers are depicted in Fig. 10. In this example, H and W are both set to 1 and dt is set to 3. Further details are given in the Sect. 2.4.

An overview of the proposed "3D FireClassNet" network is presented in Fig. 11. It consists of four main blocks, without taking into account the input and output data. Three successive blocks of convolutional layers, each accompanied by pooling layers, are intended for feature extraction. The final block consists of fully connected layers designed for video recognition tasks.

The initial block is composed of a sequence comprising a 3D convolution layer, a ReLU non-linear activation function, a batch normalization layer, and a 3D pooling layer. The second and third blocks of layers have the same structure, where the 3D convolution and non-linear layers are doubled, followed by batch normalization and 3D pooling layers. The successive layers in these three blocks serve to extract spatial and temporal features, thanks to the Conv3D and MaxPooling3D layer architecture. It's important to note that the ReLU non-linear activation function is applied after each Conv3D layer, in order to enhance the learning speed. Besides, a normalization operation is carefully added before the 3D pooling layer. This serves the dual purpose of regulating the distribution of inputs to the hidden neurons and enhancing the training speed and overall performance. After the extraction of spatial and temporal features through the three first blocks, the fourth block is dedicated to recognize fire in videos. The input of this block is a vector that reshapes the extracted features into 1D array, which is then processed by the dense layers. This block consists of a fully connected layer, with 1024 neurons, followed by a ReLU, batch normalization, and dropout layers. Subsequently, there

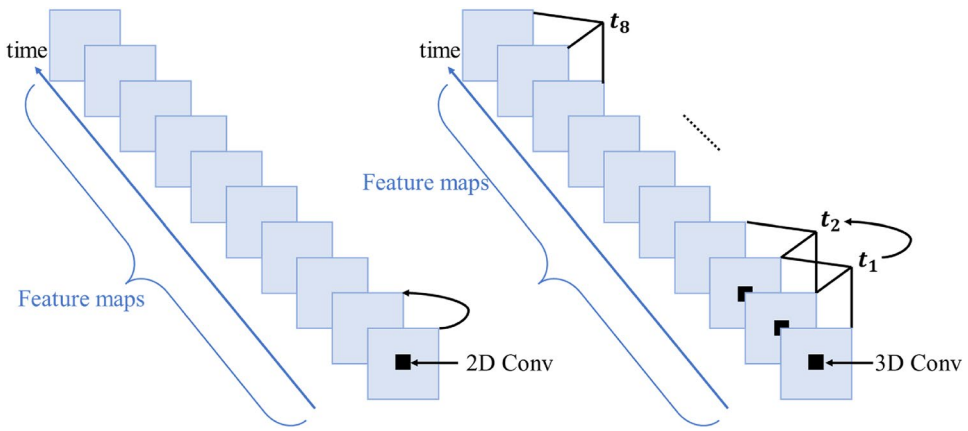

Fig. 10 2D and 3D convolutional layer structures

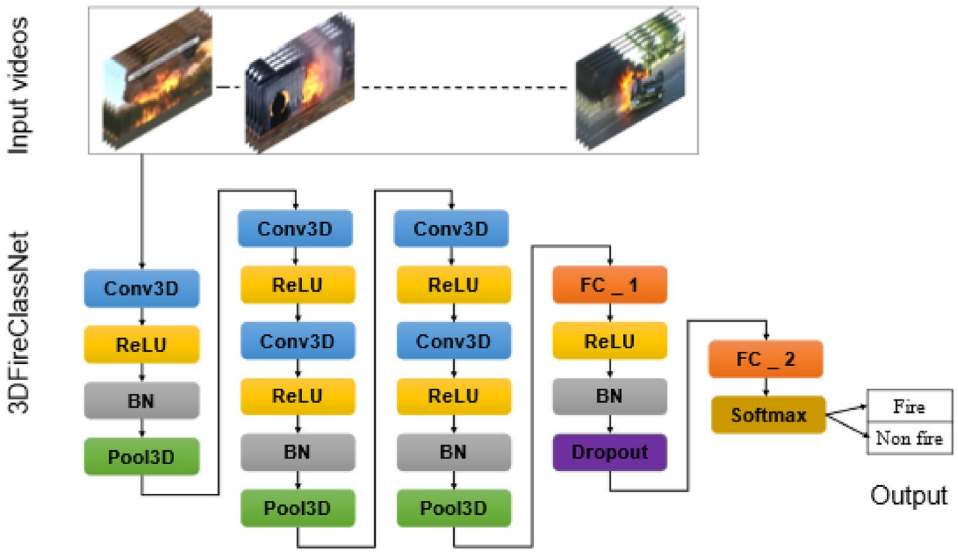

Fig. 11 Structure of the proposed "3D FireClassNet" network

is another fully connected layer with two neurons, and finally a SoftMax layer, as the constructed dataset contains two distinct classes.

In this architecture, 22 frames from each preprocessed video data, resized to 64×64, serve as inputs to the "3D FireClassNet" network. Thus, the input shape for the 3D convolution layer is (64, 64, 22, 3), where 64×64 represent the width and height of the frame. 22 corresponds to the number of selected frames in a video sequence, known as the depth hyperparameter. The 3 value is the number of channels. The initial 3D convolution layer employs 16 three-dimensional filters of size $3 \times 3 \times 3$ on the input data. All subsequent 3D convolutional layers use filter kernels with a size of $3 \times 3 \times 3$. These layers are introduced as spatio-temporal convolutional layers with a stride of 3. Every filter moves in three directions (x, y, t) to compute the feature representations, as seen in Fig. 6. The output of the 3D convolution is a feature map, represented as a 3D volume (64, 64, 22, 16). This resulting feature map is required for data analysis, spatial and temporal context. This layer contains 1312 trainable parameters. As defined, the trainable parameters of a 3D CNN structure encompass all the weights (W) and biases (B) in the network. These weights and biases constitute the two types of parameters in each layer. In the case of a 3D Conv layer, the sum of weights and biases returns the number of parameters ($Param_{Conv}$), computed via the following formula:

$$Param_{conv} = W_{conv} + B_{conv} = (K * K * K * C + 1) * Nbr_{filters} \quad (1)$$

where W_{conv} represents the number of weights ($W_{conv} = K * K * k * C * Nbr_{filters}$), with K is the dimension of the used 3D filter kernel, C is the number of channels in input images, and N denotes the filters number, also representing the biases number ($B_{conv} = N$). Table 1 provides a detailed analysis of how these trainable parameters are derived.

After the initial Conv3D layer, a MaxPooling3D layer, with a size of $2 \times 2 \times 2$, is applied. It is used to reduce the dimensions of data from an input size equal to (64, 64, 22, 16) to (22, 22, 8, 16). The output dimensions of each 3D max pooling operation follow the same equation as the resulting 2D max pooling output size, provided by Eq. (2).

Table 1 The parameters of the proposed "3D FireClassNet" network

Layer type	Output shape	Weights	Biases	Number of parameters
Input frame	(64, 64, 22, 3)	0	0	0
Conv3D	(64, 64, 22, 16)	1296	16	1312
BN	(64, 64, 22, 16)	64	0	64
MaxPool3D	(22, 22, 8, 16)	0	0	0
Conv3D	(22, 22, 8, 32)	13,824	32	13,856
Conv3D	(22, 22, 8, 32)	27,648	32	27,680
BN	(22, 22, 8, 32)	128	0	128
MaxPool3D	(11, 11, 4, 32)	0	0	0
Conv3D	(11, 11, 4, 64)	55,296	64	55,360
Conv3D	(11, 11, 4, 64)	110,592	64	110,656
BN	(11, 11, 4, 64)	256	0	256
MaxPool3D	(6, 6, 2, 64)	0	0	0
FC_1	1024	4,718,592	1024	4,719,616
BN	1024	4096	0	4096
FC_2	2	2048	2	2050
Total				4,935,074

$$Output_{size} = \left(\frac{Input_{size} - Filter_{size} + 2xPadding}{Stride} \right) + 1 \quad (2)$$

Also, the depth value is reduced from 22 to 8, using the same formula. As it is indicated in Table 1, the MaxPooling3D layer has no trainable parameters, unlike the batch normalization layer, which has 4 parameters, resulting in a total BN parameters: $Param_{BN} = 4 * N$. Consequently, the total number of parameters in this first block is 1376. It is obtained by summing the parameters of the Conv3D and BN layers (1376 = 1312 + 4 * 16).

In the second block of the "3D FireClassNet", two successive Conv3D layers are employed. Each one is with 32 filters and the same filter size of $3 \times 3 \times 3$ as the first 3D convolution layer. The number of trainable parameters for every layer in these two Conv3D layers is 13,856 and 27,680 respectively.

All the Conv3D layers use filter kernels with dimensions of $3 \times 3 \times 3$. The deployment of these two Conv3D layers results in a 3D feature map of size (22, 22, 8, 32). The next layer is obtained by employing the MaxPooling3D with a $2 \times 2 \times 2$ kernel size and a stride of 2. The resulting 3D feature maps are further reduced to a size of (11, 11, 4, 32). This second block contains a total of 41,664 parameters, as deduced from Table 1. The third block follows a similar structure to the previous one, differing primary in the used filters number. In fact, 64 filters of size $3 \times 3 \times 3$ are applied for each Conv3D layer, resulting in downsized feature maps of dimensions (11, 11, 4, 64). Behind the final 3D convolution layer, there is a MaxPooling3D with the same size $2 \times 2 \times 2$ as the previous Maxpooling3D layer. This results in a 3D feature map of size (6, 6, 2, 64), leading to 166,272 trainable parameters (the sum of the parameters number from the two Conv3D and the BN layers). Based on the analysis of this table, it is claimed that increasing the number of filters also increases the parameters number.

At the end of the network, a flattened layer is introduced, transforming the feature maps generated by the last MaxPooling3D layer into a single vector. This vector is then passed through a fully connected layer, which consists of 1024 units. In terms of parameters number, this results in a total of 4,719,616 parameters, obtained by applying this equation:

$$\begin{aligned} &Param_{FC\_1} \\ &\quad = W_{FC\_1} + B_{FC\_1} \\ &\quad = (S_{Width} * S_{Width} * S_{Height} * N + 1) * Nbr_{neurons} \end{aligned} \quad (3)$$

where S_{Width} and S_{Height} are the width and height size of the output feature map of the layer just before the fully connected layer. $Nbr_{neurons}$ is the number of neurons in the FC layer. The FC layer incorporates non-linear, batch normalization, and dropout techniques. The non-linearity is intended to accelerate the learning process in the fourth block for the recognition task. Batch normalization serves to normalize the previous layer's values in each batch. As a form of regularization, dropout is applied, in our "3D FireClassNet", with the dense layer, specifically before the last FC layer. The standard dropout probability of 0.5 is used, in order to reduce more the issue of overfitting. The final dense layer comprises 2 neurons designed for the output prediction. Its size is selected to be equal to the number of target classes (fire and non-fire). This layer ends with a SoftMax classifier, defined by the following Eq. (4):

$$P(z_i) = \frac{e^{z_i}}{\sum j^{Nbr\ of\ classes} e^{z_i}} \quad (4)$$

This ensures a probabilistic representation for the different classes. For a fire video recognition task with 2 classes, the number of parameters in the final fully connected layer is 2050. It is computed using Eq. (5), which is distinct from formula (3). This difference is due to the existence of two types of fully connected layers in a CNN. The first FC type is connected to the last Conv3D or MaxPooling3D layers, whereas the second one is connected to other FC layers.

$$Param_{FC\_2} = W_{FC\_2} + B_{FC\_2} = ((Nbr_{neurons})_{-1} + 1) * Nbr_{neurons} \quad (5)$$

Here, $(Nbr_{neurons})_{-1}$ denotes the number of neurons in the previous FC layer. Therefore, the total number of trainable parameters in the presented "3D FireClassNet" model is 4,935,074, where all of them are initialized randomly. It is obtained by summing the parameters number of all 3D convolutional, BN, and FC layers.

5 Experimental evaluation and discussion

To assess the effectiveness of the proposed 3D fire recognition model, the experimental protocol is firstly introduced in Sect. 5.1. It outlines the description of the constructed dataset, used for the training and validation of the proposed network, as well as the experimental architecture fine-tuning. Then, Sect. 5.2 focuses on validating the model using different numbers of filters and the adopted experimental protocol. In this subsection, experiments are presented, and their results are analyzed and discussed. Furthermore, Sect. 5.3 provides a comparative study between the proposed "3D FireClassNet" and the other related works, particularly the hand-crafted approaches and the developed spatio-temporal architectures. A comparison between our designed 3D CNN and the 2D network suggested in Daoud et al. (2023) is presented in Sect. 5.4.

It is noteworthy that the training of the derived 3D model is carried out using the deep learning framework Keras on the following hardware specifications: Intel (R)

Core (TM) i5 2.5 GHz, 8 GB RAM and Nvidia Geforce GTx 980 M 2 GB GPU.

5.1 Experimental protocol

5.1.1 Dataset description

For the different experiments, we try to create our own dataset for the "3D FireClassNet" training, since there is no standard fire database for this research field. This dataset is constructed by collecting the available video sequences from the public databases (Cetin 2007; Foggia et al. 2015; Cazzolato et al. 2017; Grammalidis et al. 2017; Phillips Iii et al. 2002; Steffens et al. 2015). We ensure a more diverse representation of fire events on different samples. This dataset consists of two distinct categories: fire and non-fire video sequences. All details are displayed in Table 2. This established dataset comprises 92 fire scenes, and 38 non-fire sequences. The initial step in the training process of the "3D FireClassNet" network, involves loading data to be passed through the first network layer. To achieve this, 22 successive frames are randomly taken from each video sequence, since the depth hyperparameter is set to 22. This value is experimentally determined, and it is adopted based on the analysis and discussions, presented in Sect. 5.2.2. Selecting successive frames allows us to explore the temporal features present in video sequences.

As indicated in the preprocessing phase (in Sect. 4.1), our main goal is to enlarge the constructed dataset to address the challenge of limited samples and lack of diversity. Hence, three transformations, including rotation, horizontal and vertical flipping, are applied to the data. This results in a total of 520 video sequences, comprising 11,440 frames, forming our created dataset that will be used as inputs for the "3D FireClassNet". By employing video data augmentation techniques, we enhance the dataset's diversity and make it more challenging. These transformations ensure, in the video frames, various positions of fire and other objects.

5.1.2 Training and validation of the 3D model

As mentioned previously, once the network structure is developed, the subsequent critical phase involves training and validation of the model in order to evaluate its performance. For this purpose, the constructed dataset is partitioned into three distinct groups: 60% of the data is allocated for training, 20% is for validation and the remaining 20% is set aside to test the model's effectiveness on a holdout dataset. This partition strategy is designed to ensure an equitable assessment of the model's capacity to generalize and its competence in handling unseen data. Subsequently, in order to evaluate and compare the performance of the developed model against other spatio-temporal deep learning models designed for fire recognition in videos, various datasets are employed.

Table 2 Description of the dataset used for the 3D model training

	Number of fire videos	Number of non-fire videos	Total
Collected videos	92	38	130
Augmented videos	368	152	520

5.1.3 Experimental architecture tuning

For the optimization of the presented deep fire recognition model, hyperparameters of the "3D FireClassNet" has to be carefully selected for an effective model. The adopted hyperparameters are listed in Table 3. Experiments are carried out for different depths of the "3D FireClassNet" network: depth = 8, 10, 20, 22, 25. The depth dimension represents the temporal dimension. It is the number of frames taken from each input video to construct the 3D volume of data. Fixing the depth is necessary to accurately capture the temporal information and ensure that the length of the input videos is the same. For that, every experiment is repeated by varying the depth value to validate our adopted choice, to 22 frames. Moreover, the number of epochs is determined, using the same method applied for the selection of temporal dimension value. In fact, the number of epochs is varied to 50, 100, 200, and 300. Then, the value that yields to the best results is retained, which is 200 epochs. The impact of the depth and the epochs number on the 3D model's performance is further analyzed in the subsequent subsections. Additionally, the number of training frames in a single set, defined by the batch size hyperparameter, is set to 8 for our network training.

During the training process, the optimizer plays a significant role in enhancing the reduction of loss and updating

Table 3 Training hyperparameters of the proposed "3D FireClassNet" network

Hyperparameter	Value
Depth	22
Epochs	200
Batch size	8
Optimizer	RMSProp
Weights initialization	Randomly
Initial learning rate	Initial Lr = 0.001
Learning rate tuning method	The learning rate is reduced by
	A factor of 0.5 each 50 epochs

of the network's parameters. For that, various available optimizers, such as Stochastic Gradient Descent (SGD), Adaptive Gradient (AdGrad), Adaptive Moment Estimation (Adam), and Root Mean Square Propagation (RMSProp) are tested to select the most adequate one for the "3D FireClassNet" training. Based on the results analysis, RMSProp improves the model's performance. Thus, it is retained for our network training. To further optimize our 3D CNN, random initialization of the weights is applied. Obviously, randomly initializing parameters consistently yields to superior results in our experiments, when compared to alternative initialization methods.

Since the learning rate directly determines the network's efficiency by updating the weights and parameters, it is considered as the most crucial hyperparameter, which should be carefully tuned. Consequently, different initial learning rates are tested to investigate the effect of the learning rate on the "3D FireClassNet" network's performance. The initial learning rate is first set to 0.1. Then for each test experiment, it is set to 0.001, and 0.0001, respectively. According to the experimentation, the best value for finding a good local minimum, is 0.001. Then, for each 50 epochs, the learning rate is reduced by a factor of 0.5. Using this learning rate fine-tuning method, as illustrated in Fig. 12, results in lower local loss and increased accuracy.

To further optimize the designed network, several additional parameters are incorporated into the training process. These parameters, including the data augmentation technique, batch normalization, and dropout, are also adopted. Indeed, data augmentation is used to expand the variability of samples in the dataset. This is achieved by applying three types of transformations to the input videos: rotation, horizontal, and vertical flipping. These transformations generate diverse representations of the frames, featuring various orientations and positions of the fire object. As a result, this approach mitigates the limitations posed by the relatively small number of video frames.

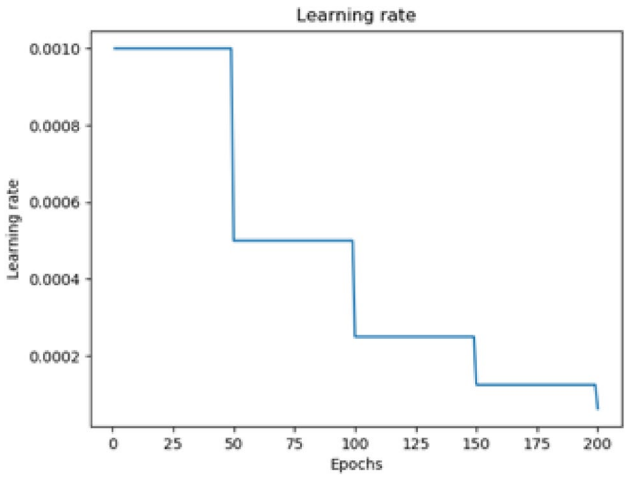

Fig. 12 The learning rate tuning method during the training with an initial value of 0.001

Additionally, batch normalization is also implemented in the "3D FireClassNet" network, specifically after the non-linear layer. Batch normalization is used to standardize neuron activations across layers and replaces bias variables (Nithya et al. 2024). This leads to improved model efficiency and training speed.

Another regularization technique applied to our 3D CNN is the dropout, with a probability of 0.5. It is primarily utilized after the final fully connected layer. Dropout is employed to address a fundamental issue that can impact the model's predictability, called overfitting. During training, this technique randomly drops units from the network, enhancing the model's generalization. The fine-tuning of the presented hyperparameters and the judicious selection of parameter configurations contribute to the improved performance of our proposed network.

5.2 Evaluation of the "3D FireClassNet" architecture

The efficacy of the presented "3D FireClassNet" network is assessed by analyzing the training and validation accuracies (Acc) and the loss function (Loss), introduced in Eqs. 6 and 7.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{6}$$

$$Cross\_Entropy\ Loss = -\sum_{i}^{C} Y_i * log\hat{Y}_i \tag{7}$$

The accuracy value indicates the percentage of the correct recognized fire scenes. The loss function, computed via cross-entropy, is the metric reduced by the optimizer during the model training. It calculates the difference between the model's predictions (represented as Y_i in Eq. 7) and the target labels that we aim to predict, known as ground truth (represented as Y_i in Eq. 7), for each class i in the set of classes C. In the specific fire image classification task, C contains two classes: fire and non-fire. During each phase of training, validation, and testing, the network's accuracy can be measured and monitored to demonstrate how quickly it is improving. By minimizing the cross-entropy loss, the model adjusts its parameters to improve its accuracy in distinguishing between fire and non-fire classes.

In addition, other evaluation metrics, including recall (R), precision (P), and F1 score ($F1$) are computed to evaluate the performance of our 3D CNN. These metrics are obtained using the following equations:

$$Recall = \frac{TP}{TP + FN} \tag{8}$$

$$Precision = \frac{TP}{TP + FP} \tag{9}$$

$$F_1 score = 2 * \frac{Recall * Precision}{Recall + Precision} \tag{10}$$

where

- TP represents the number of fire video sequences that are correctly recognized as fire;
- FP denotes the number of non-fire videos which are wrongly recognized as fire;
- TN is the count of non-fire scenes correctly recognized as non-fire;
- FN indicates the number of fire video sequences that are incorrectly recognized as non-fire scenes.

Aiming to validate our novel "3D FireClassNet" network structure, two architectures with a variation in the number of filters are proposed. Their results are analyzed and discussed to validate the adopted "3D FireClassNet". Furthermore, our model is evaluated based on the retained hyperparameters and the used parameters. The outcomes of these assessments and the conclusion drawn are presented and discussed in the subsequent subsections.

5.2.1 Validation of the filters number

The number of filters in the convolutional layers is defined as follows: 16 filters are applied in the first 3D convolution of the first block, they are then doubled to 32 in every 3D convolutional layer of the second block, and it is finally increased to 64 in each 3D convolutional layer of the third block. All these filters use a kernel size of $3 \times 3 \times 3$. In order to demonstrate the impact of the filters number on the fire recognition model's performance, it is suggested to build a second variant of our architecture, where the number of filters is reduced by keeping the kernels size to $3 \times 3 \times 3$. A detailed description of the two variants of our architecture is provided in Table 4. The first block starts with 8 filters in the Conv3D layer. Then, it is increased to 16 filters in the convolution layers of the second block, to reach 32 filters in the two Conv3D layers of the third block. The results of these two architectures training are exhibited in Table 5.

It is demonstrated, from Table 5, that the second proposed network, which employs a reduced number of filters, outperforms the "3D FireClassNet" network throughout the training and validation processes. The second network achieves a high training accuracy (Acc_{Train}) and a lower loss value ($Loss_{Train}$). During the validation phase,

Table 4 Details of the "3D FireClassNet" network structure and the second variant of the proposed architecture

Type	Input	Filter	Kernel size	Output
1st Network "3D FireClassNet"				
Conv3D	(64, 64, 22, 3)	16	$3 \times 3 \times 3$	(64, 64, 22, 16)
MaxPool3D	(64, 64, 22, 16)	16	$3 \times 3 \times 3$	(22, 22, 8, 16)
Conv3D	(22, 22, 8, 16)	32	$3 \times 3 \times 3$	(22, 22, 8, 32)
Conv3D	(22, 22, 8, 32)	$323 \times 3 \times 3$	(22, 22, 8, 32)	
MaxPool3D	(22, 22, 8, 32)	32	$2 \times 2 \times 2$	(11, 11, 4, 32)
Conv3D	(11, 11, 4, 32)	64	$3 \times 3 \times 3$	(11, 11, 4, 64)
Conv3D	(11, 11, 4, 64)	64	$3 \times 3 \times 3$	(11, 11, 4, 64)
MaxPool3D	(11, 11, 4, 64)	$642 \times 2 \times 2$	(6, 6, 2, 64)	
FC_1	4608	–	–	1024
FC_2	1024	–	–	2
2nd network				
Conv3D	(64, 64, 22, 3)	8	$3 \times 3 \times 3$	(64, 64, 22, 8)
MaxPool3D	(64, 64, 22, 8)	8	$3 \times 3 \times 3$	(22, 22, 8, 8)
Conv3D	(22, 22, 8, 8)	16	$3 \times 3 \times 3$	(22, 22, 8, 16)
Conv3D	(22, 22, 8, 16)	16	$3 \times 3 \times 3$	(22, 22, 8, 16)
MaxPool3D	(22, 22, 8, 16)	16	$2 \times 2 \times 2$	(11, 11, 4, 16)
Conv3D	(11, 11, 4, 16)	32	$3 \times 3 \times 3$	(11, 11, 4, 32)
Conv3D	(11, 11, 4, 32)	32	$3 \times 3 \times 3$	(11, 11, 4, 32)
MaxPool3D	(11, 11, 4, 32)	32	$2 \times 2 \times 2$	(6, 6, 2, 32)
FC_1	2304	–	–	1024
FC_2	1024	–	–	2

Table 5 Training results of the twoproposed 3D CNN networks when varying the number of filters

	Training		Validation		Testing		Parameters
	Acc_{Train} (%)	$Loss_{Train}$	Acc_{Val} (%)	$Loss_{Val}$	Acc_{Test} (%)	$Loss_{Test}$	
1st network "3D FireClassNet"	99.36	0.0088	98.1	0.0380	98.077	0.0696	4,935,074
2nd network	100	1.25 e−07	99.04	0.0535	96.15	0.1064	2,419,282

the accuracy remains high at 99.04%, whereas the "3D FireClassNet" reaches an accuracy rate (Acc_{Val}) of 98.1% with a lower loss value of 0.0380. A comparison of these values is also represented in Fig. 13. The training and validation accuracy curves (Fig. 13a, c) show the two models' progress during the training and validation processes. In Fig. 13a, the validation curve of the "3D FireClassNet" is close to the training curve, which converges to 1. But in Fig. 13c, the validation curve of the second network has multiple oscillations, which is different to the "3D FireClassNet" validation curve. Whereas the training and validation loss curves (Fig. 13b, d) exhibit a decrease to a converging point, with a minor difference in their final values. These results confirm the model's capability to

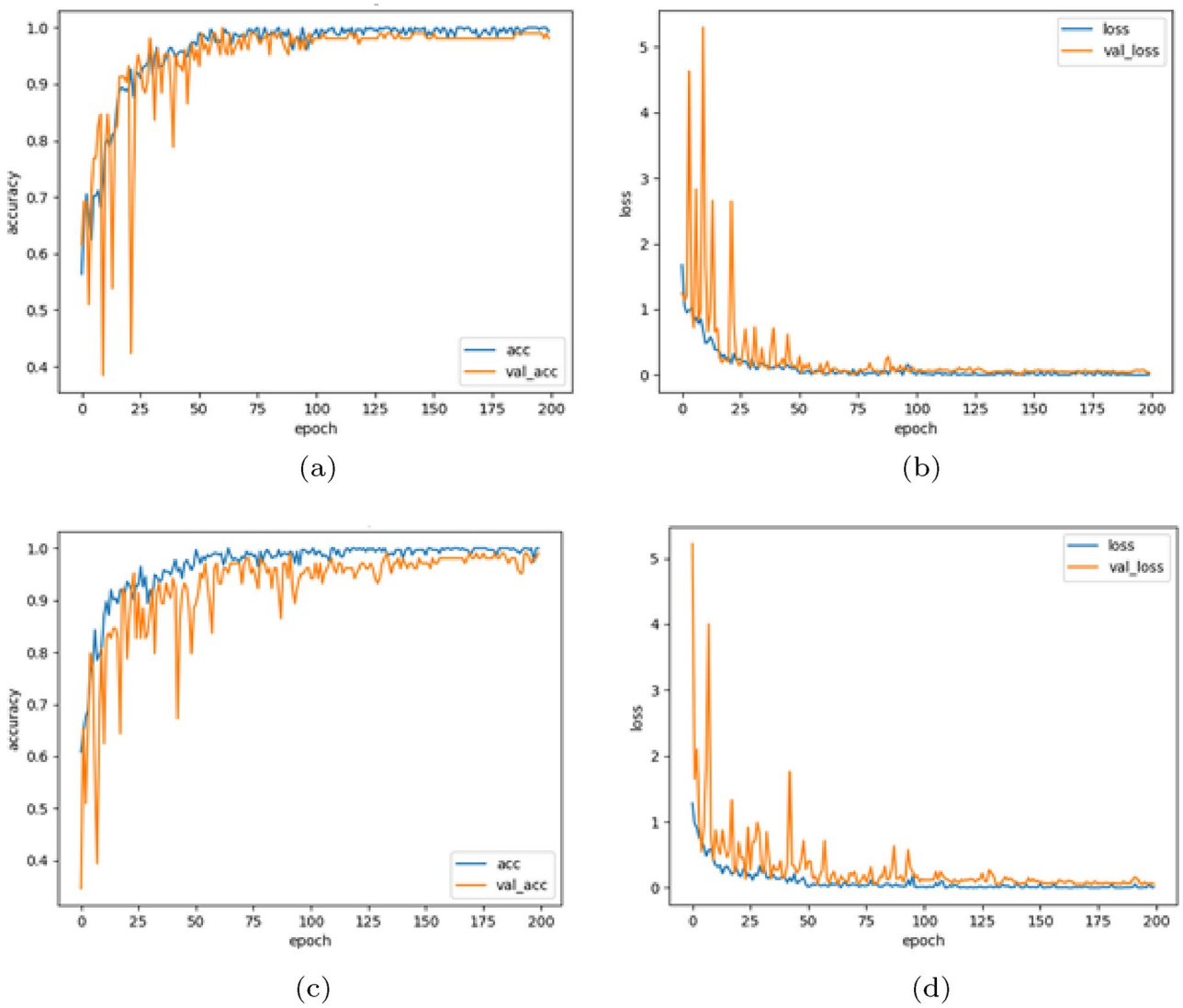

Fig. 13 Training and validation results of the two proposed networks: **a** and **b** accuracy and loss curves of the "3D FireClassNet", **c** and **d** accuracy and loss of the second network

accurately predict unseen data samples. However, the highest values of accuracy and the lowest loss are obtained during the testing phase, by the "3D FireClassNet". This demonstrates the first proposed network's ability to effectively recognize the test video data. In experimental terms, the results are very close, with only a slight difference. Visually, the representations of the training and validation curves of the "3D FireClassNet" are closer than those of the second proposed network. Based on these observations and the resulting (Acc_{Test}) and ($Loss_{Test}$) values, the structure of the "3D FireClassNet" is adopted. This is despite the number of parameters and the training time that are relatively higher, compared to the second network.

To further justify our choice of the "3D FireClassNet" network structure, the two derived models are evaluated. Their performance is depicted in Table 6. The number of well recognized videos (TP and TN) is the highest with the "3D FireClassNet". Similarly, for FN, just two fire scenes are wrongly recognized as non-fire videos, compared to four non-fire scenes with the second network. This reflects the improved value of recall in the first proposed network, which reaches 99.46% compared to 98.91% of the second one. Additionally, the accuracy value 99.23%, is superior to the rate of 99.038% achieved by the second model. These findings prove the ability of the developed model, generated by the "3D FireClassNet" training, to accurately recognize fire video sequences. Nevertheless, the resulting precision rate of the second variant of our architecture is better with a difference of 0.27%. This is since two non-fire videos are incorrectly recognized as fire. Thus, the number of false positives should be further reduced to enhance the model's performance. Based on these outcomes, the model derived by "3D FireClassNet" network performs better than the second model, obtained by training the proposed variant of our architecture. This further justifies our selection of the presented "3D FireClassNet".

5.2.2 Experimental hyperparameters tuning for an enhanced "3D FireClassNet" architecture

In order to validate the retained hyperparameters, several distinct models are trained to validate the effectiveness of our novel model in recognizing fires in video sequences. To achieve this and to confirm our selection of the experimental protocol for the suggested 3D CNN network optimization, one of the hyperparameters is varied, while keeping the others constant.

- **Validation of the learning rate tuning method** During the development of the 3D fire recognition model, parameters are updated based on the choice of hyperparameters, particularly the learning rate. It significantly influences the performance of the network. That's why it is proposed to evaluate the impact of different initial learning rates on the network's performance. Initially, a learning rate of 0.01 is tested. Then, the learning rates for every trial are set to 0.001 and 0.0001, respectively. The experimental findings are detailed in Table 7. For an initial learning rate equal to 0.01, the model achieves an accuracy of 97.11% on the testing set, which is 0.967% lower compared to the accuracy reached with an initial learning rate of 0.001. When the initial learning rate is set to 0.0001, the testing accuracy (Acc_{Test}) drops to 91.35%, with a reduction of 6.727% compared to the accuracy at 0.001. These results are also observed in the loss values. Thereby, the model, with an initial learning rate of 0.001, performs well, achieving the highest testing accuracy of 98.077% and the lowest loss value of 0.0696. Thus, the chosen initial learning rate for our novel 3D model is 0.001. From these results, the effect of the initial value of the learning rate on the "3D FireClassNet" training is shown, demonstrating that the designed model is effective at recognizing fire and non-fire sequences on unseen data (testing set). In order to avoid the issues of overfitting or underfitting, the learning rate has to be adjusted and monitored. So it is reduced by a factor of

Table 6 The performance of the two proposed 3D CNN when varying the number of filters

Proposed networks	TP	FN	FP	TN	P (%)	R (%)	F1 (%)	A (%)
"3D FireClassNet"	**366**	**2**	2	**150**	99.46	**99.46**	**99.46**	**99.23**
2nd network	364	4	**1**	151	**99.73**	98.91	99.32	99.038

Table 7 Accuracy and loss results of the proposed network "3D FireClassNet" by varying the initial value of the learning rate

Initial value	Training		Validation		Testing	
	Acc_{Train} (%)	$Loss_{Train}$	Acc_{Val} (%)	$Loss_{Val}$	Acc_{Test} (%)	$Loss_{Test}$
0.01	99.04	0.1423	98.08	0.3099	97.11	0.3956
0.001	**99.36**	**0.0088**	**98.1**	**0.0380**	**98.077**	**0.0696**
0.0001	99.35	0.0122	94.23	0.2815	91.35	0.5264

Table 8 Effect of the learning rate tuning method

	Training		Validation	
	Acc_{Train} (%)	$Loss_{Train}$	Acc_{Val} (%)	$Loss_{Val}$
0.001	96.79	0.0874	96.15	0.1026
0.0005	99.04	0.0231	98.07	0.0713
0.00025	98.72	0.0478	98.08	0.0470
0.000125	99.36	0.0088	98.1	0.0380

0.5 for each 50 epochs, as indicated in Table 3 and presented in Fig. 12. The impact of this learning rate tuning method is presented in Table 8. The learning rate is dropped from an initial value of 0.001 to a final value of 0.000125. With each reduction by 0.5, both training and validation accuracies are enhanced. The Acc_{Train} is increased from 96.79% to reach 99.36%. And the training loss is decreased significantly from 0.0874 to 0.0088. These improvements are also observed in the validation process. During these two processes of training and validation, decreasing the learning rate enhances the model stability and allows for finer adjustments to the weights. This makes the 3D model converges to a more optimal point in terms of loss.

- **Validation of the depth hyperparameter** Different experiments are conducted to evaluate the impact of the depth hyperparameter on the model's performance. As indicated in Sect. 5.1.3 of this Chapter, the depth hyperparameter is the temporal dimension of the "3D Fire-ClassNet". It represents the quantity of extracted frames from each input video to create the three-dimensional data volume. The depth is firstly set to 8, and then to 10, 20, 22 and 25 for each trial. The results of these variations are exhibited in Table 9. It is shown that when the depth is equal to 8 and 25, the accuracy of the "3D FireClassNet" over the training set is at 100%. However, the testing accuracies reach 96.15% and 88.5% respectively. The high value of accuracy on the training set and the low accuracy on the testing set reflect that the model is overfitting. The highest testing accuracy (Acc_{Test}) of 98.077% is obtained when the depth hyperparameter is set to 22, and this leads to closer values for Acc_{Train} and Acc_{Val}. Add to that, the lowest loss values are achieved when the depth is equal to 22. Therefore, a depth value of 22 is adopted. These outcomes are also confirmed by Fig. 14, illustrating the effect of varying the depth value on the model's performance. The highest values, achieved when the depth is set to 22, prove the model's robustness, compared with the other obtained models. Our 3D presented model achieves 99.46% in terms of recall and precision respectively. A high recall value indicates that the majority of fire video sequences are correctly recognized, resulting in a high number of true positives and a low number of false negatives. Furthermore, the reached precision is 99.46%, reflecting a low number of false positives (only two non-fire videos are incor-

Table 9 Accuracy and loss results of the proposed network "3D FireClassNet" when varying the depth hyperparameter

Depth	Training		Validation		Testing	
	Acc_{Train} (%)	$Loss_{Train}$	Acc_{Val} (%)	$Loss_{Val}$	Acc_{Test} (%)	$Loss_{Test}$
8	**100**	0.0185	**99.03**	0.0586	96.15	0.1837
10	99.67	0.0163	96.15	0.1219	92.31	0.3865
20	99.6	0.0165	94.23	0.2748	92.30	0.4636
22	99.36	**0.0088**	98.1	**0.0380**	**98.077**	**0.0696**
25	**100**	0.0082	88.46	0.2886	88.5	0.5584

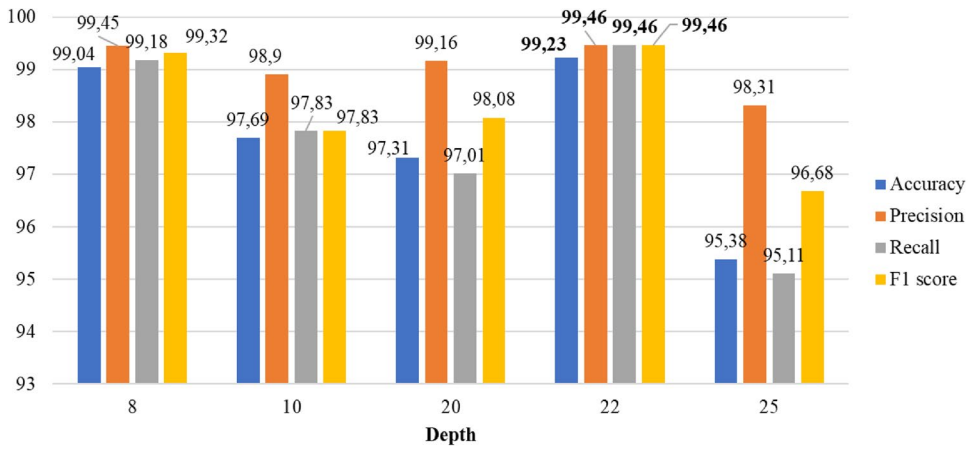

Fig. 14 Test performance of the proposed model when varying the depth value

rectly classified as fire scenes). Figure 15, displaying the various confusion matrices, further justifies these results. In conclusion, our model, with a temporal dimension in the input vector, demonstrates its ability to accurately identify most fire and non-fire videos with a minimal number of false positives.

- **Validation of the used optimizer** Further experiments are carried out to validate the choice of adopted hyperparameters, specifically the optimizer and the use of data augmentation technique. As previously mentioned, one hyperparameter is varied while the others are fixed. As depicted in Table 10, four types of optimizers are tested: Adam, SGD, AdaGrad, and RMSProp. Then, two trials are performed, one without data augmentation and the other with data augmentation, to demonstrate the added value of this technique in improving the model's efficiency. By analyzing the results, the derived models obtained by the Adam and SGD optimizers achieve higher training accuracies, compared to their testing accuracies. When using the Adam optimizer, the Acc_{Train} is 100% which is 2.89% greater than the testing accuracy (Acc_{Test}). This may lead to the possibility of overfitting issue. Similarly with the SGD and AdaGrad optimizers, the testing accuracies (Acc_{Test}) are 86.53% and 84.61% respectively, which are 13.47% and 15.39% lower from the training accuracies (Acc_{Train}). This gap between the training and testing accuracies is not observed when using the RMSProp optimizer. In fact, the RMSProp optimizer yields to higher training results, compared to other optimizers. These findings are approved in Fig. 16, which shows the impact of varying the optimizer on the model's performance. As depicted, RMSProp outperforms the other optimizers in terms of accuracy and F1 score. The main issue with AdaGrad is its slow convergence because the sum of squared gradients only accumulates and never decreases. RMSProp addresses this problem by introducing a decay factor, which allows the accumulated sum to gradually reduce over time, preventing the learning rate from becoming too small. Similarly, the limitation of the SGD optimizer is its slower convergence. For the Adam, results are close to those of RMSProp, without exceeding them. The superior performance of RMSProp reflects its ability to stabilize the learning process by dividing the learning rate by the moving average of the squared gradients. This results in a more consistent training process. With the RMSProp optimizer, our derived 3D model achieves the highest precision and recall, indicating the high level of accuracy and F1 score. This means that our model correctly identifies most fire video sequences, yielding to a high number of true positives and a low number of false negatives. Therefore, we adopt the RMSProp optimizer for the optimization of our "3D FireClassNet".
- **Validation of the used weights initialization method** The initialization of layer weights is an important aspect in deep learning which can enhance or decrease the model's performance (Chandra and Das 2023a, b, 2024). In our experiments, different techniques including Glorot (Xavier) (Glorot and Bengio 2010), He distributions (He et al. 2015), and random initialization are tested to show their impact on the training process. The Glorot initializer depends on the input and output neurons, providing two initialization methods (Normal and Uniform). For Glorot Normal initialization, the initial value is generated from a normal Gaussian distribution centered on 0, where each layer has different standard deviation value defined by: $std = \sqrt{2/(neuron_{in} + neuron_{on})}$. In the case of Glorot Uniform initialization, the initial value is generated from a uniform distribution within [-limit, limit]. *limit* is computed by: $limit = \sqrt{(6/(neuron_{in} + neuron_{on})}$. The He initializer is proposed by He et al. because the Glorot initializer is not effective with the ReLU activation function. The He Normal initialization generates the initial value from normal Gaussian distribution centered on 0, depending only on the input units, as follows: $std = \sqrt{(2/(neuron_{in}))}$. For He Uniform initialization, weights are generated from a uniform distribution within [-limit, limit], where the limit is computed as: $limit = \sqrt{(6/(neuron_{in}))}$. Lastly, in Random initialization, weights are initialized using random values derived from Gaussian distributions. The resulting accuracy and loss values from these initializers are exhibited in Table 11. It is demonstrated that the Glorot Normal initialization technique outperforms the other initializers in each phase. During training, it achieves an accuracy (Acc_{Train}) of 99.35%. Even for validation, the accuracy (Acc_{Val}) reaches 99.04%, with a similar testing accuracy Acc_{Test} of 99.04%. These close accuracy values prove the model's ability to make accurate predictions on unseen samples during training. These values highlight the effectiveness of the Glorot Normal initialization method. In comparison to the Glorot Uniform initializer, the loss values ($Loss_{Train}$ and $Loss_{Test}$) are still higher than those obtained with random initialization. Concerning the He distributions, there is a gap between the training and testing accuracies, leading to inaccurate recognition of fire scenes. For the model initialized with random weights, the accuracy and loss values are approximately similar, showing good performance in recognizing the test data. Based on the observation that the training accuracy of the random weights' initialization method converges rapidly to a constant value, this final initializer is retained. Add to that, the accuracy and loss tend to converge faster during the validation phase, indicating a more stable training and validation processes compared to models initialized with the Glorot and He distributions.

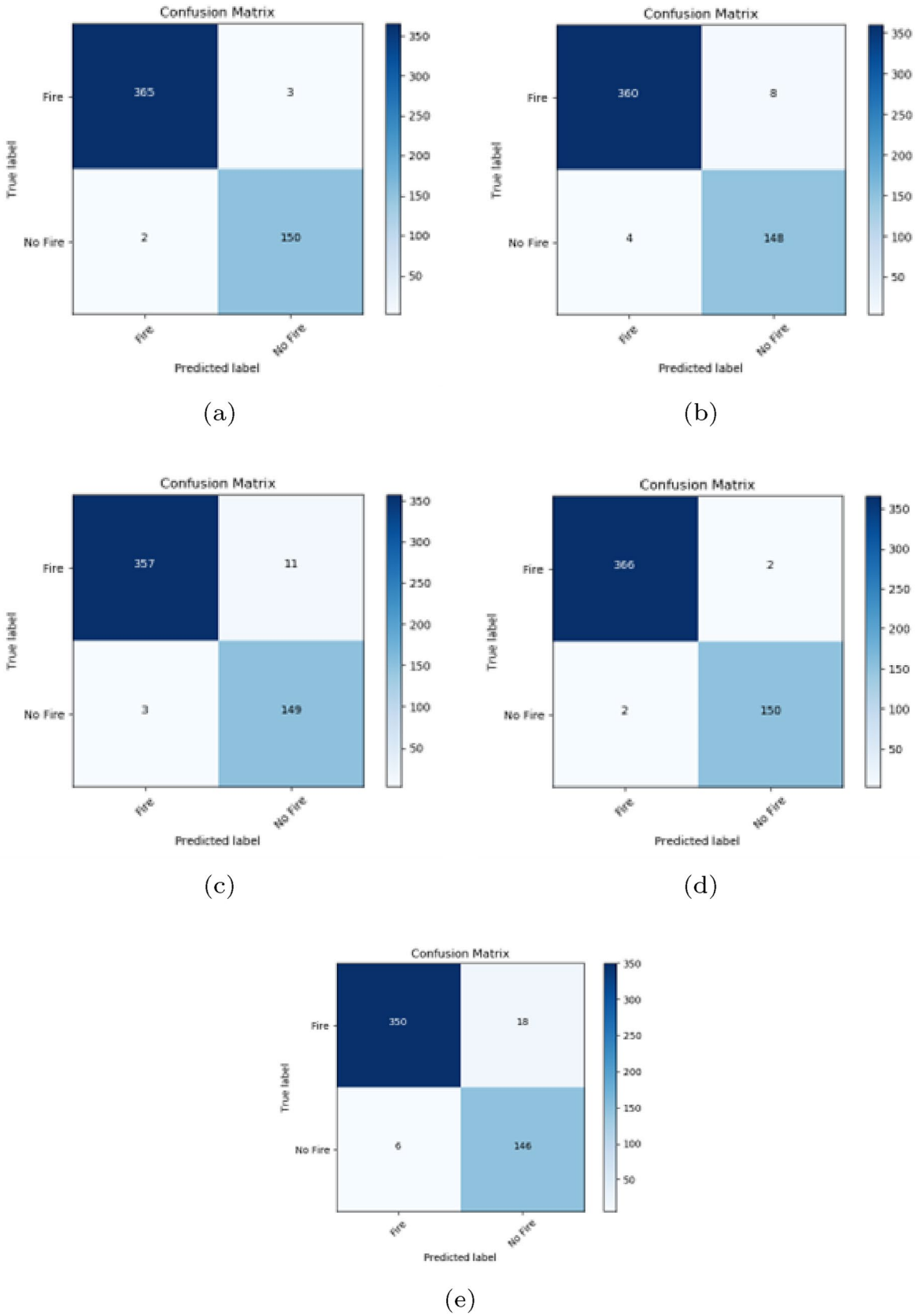

Fig. 15 The resulting confusion matrices when varying the depth hyperparameter value: **a** depth = 8, **b** depth = 10, **c** depth = 20, **d** depth = 22, **e** depth = 25

Table 10 Accuracy and loss results of the proposed "3D fireClassNet" when varying the optimizer

Optimizer	Training		Validation		Testing	
	Acc_{Train} (%)	$Loss_{Train}$	Acc_{Val} (%)	$Loss_{Val}$	Acc_{Test} (%)	$Loss_{Test}$
Adam	**100**	0.0123	**99.04**	**0.0242**	97.11	0.07666
SGD	**100**	0.02495	94.23	0.1647	86.53	0.3148
AdaGrad	**100**	0.0114	90.38	0.2383	84.61	0.3472
RMSProp	99.36	**0.0088**	98.1	0.0380	**98.077**	**0.0696**

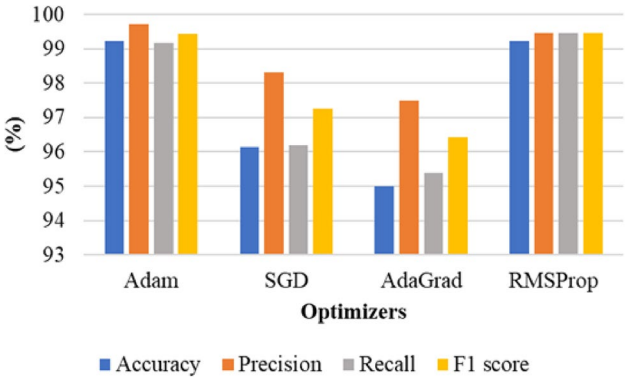

Fig. 16 Test performance of the proposed model when varying the optimizer

- **Validation of the use of data augmentation technique** Two experiments are performed, one without data augmentation and the other with data augmentation, to demonstrate the effect of video data augmentation technique on the model's enhancement. In this experiment, the "3D FireClassNet" is trained without data augmentation, using RMSProp as the optimizer, with the adopted learning rate method, and the depth value which is set to 22. The results in Table 12 indicate that, when the model is trained without data augmentation, it achieves a final test accuracy of 65.38%. However, when data augmentation technique is incorporated during training, the model's test accuracy significantly increases to 98.077%. These outcomes represent a significant improvement of 32.697%, highlighting the necessity and benefits of the augmented data in the model prediction process. The loss value ($Loss_{Test}$) is also greatly enhanced. Moreover, data augmentation improves the training and validation processes, as shown in Fig. 17. Figure 17c, d illustrate that the model trained without data augmentation strongly overfits the training data. This justifies the gap between the curves, indicating that the model performs better on training data, compared to the validation data. Hence, the divergence between the curves serves as a clear indicator of an overfitting model. Nevertheless, in Fig. 17a, b, where the "3D FireClassNet" is trained with data augmentation, the training and validation curves of the accuracy and loss are much closer. Data augmentation, which effectively increases the dataset size by generating different realistic scenes, acts as a regularization technique. It helps at preventing overfitting and enhances the model's ability to generalize and successfully predict the testing set of data. Consequently, our model's performance on unseen data is improved, which explain the improvement in the testing results (Acc_{Test} and $Loss_{Test}$). By these two experiments, the added value of data augmentation technique in improving the model's efficiency is demonstrated.

Table 11 Accuracy and loss results of the proposed network "3D FireClassNet" when varying the weights initialization method

Weights initialization method	Training		Validation		Testing	
	Acc_{Train} (%)	$Loss_{Train}$	Acc_{Val} (%)	$Loss_{Val}$	Acc_{Test} (%)	$Loss_{Test}$
GlorotNormal	99.35	0.0190	99.04	0.0242	**99.04**	**0.0575**
GlorotUniform	99.36	0.0190	**100**	**0.0005**	98.08	0.2198
HeNormal	**100**	$\mathbf{1.27 \times 10^{-7}}$	98.07	0.0321	95.19	0.0857
HeUniform	**100**	1.34×10^{-7}	95.19	0.2324	94.23	0.2379
Random initialization	99.36	0.0088	98.1	0.0380	98.077	0.0696

Table 12 Training results when adding video data augmentation over 200 epochs with RMSProp optimizer

	Training		Validation		Testing	
	Acc_{Train} (%)	$Loss_{Train}$	Acc_{Val} (%)	$Loss_{Val}$	Acc_{Test} (%)	$Loss_{Test}$
Without DA	97.37	0.1841	76.92	1.993	65.38	2.5455
Proposed with DA	99.36	**0.0088**	98.1	0.0380	**98.077**	**0.0696**

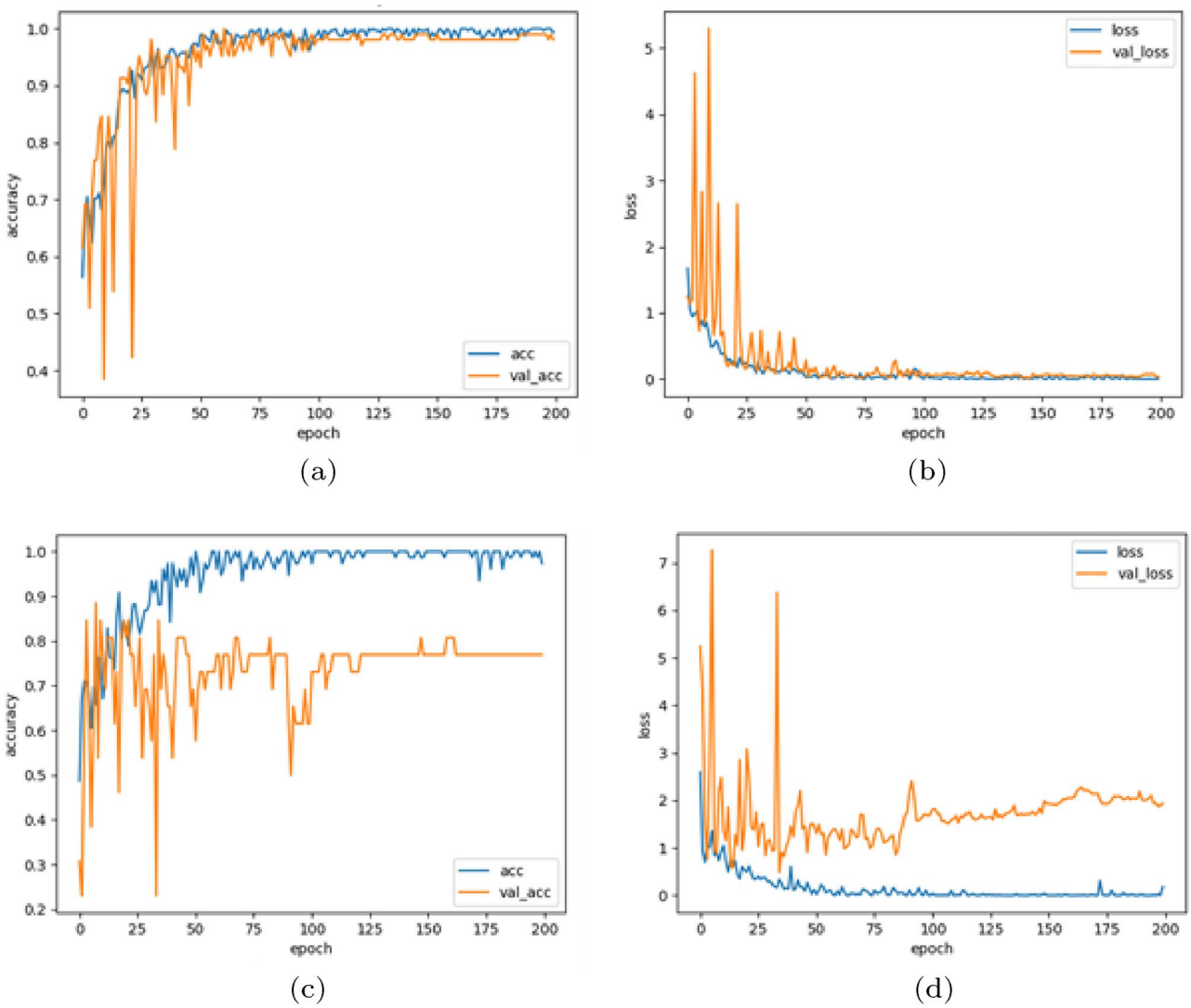

Fig. 17 Effect of adding the video data augmentation on the training and validation results of the proposed "3D FireClassNet": **a**, **b** accuracy and loss curves when training with video data augmentation, **c**, **d** accuracy and loss curves when training without video data augmentation

- **Validation of the number of epochs** Once the hyper-parameters mentioned above are selected and evaluated, the number of epochs is varied to determine the adequate one for the training of our "3D FireClassNet". It is initially set to 50, and then it is varied to 100, 200, and 300 epochs. The results of this experiment are presented in Table 13. By analyzing this table, the highest accuracy value, closest to the validation accuracy, and with the minimal losses ($Loss_{Train}$) and ($Loss_{Val}$), are achieved when the epochs number is equal to 200. Increasing this number for more training iterations may lead to an over-fitting, as observed when using 300 epochs. A difference of 8.65% is observed between the training and testing accuracy. Compared with the remaining results, using

Table 13 Accuracy and loss results of the proposed network "3D FireClassNet" when varying the number of epochs

Number of epochs	Training		Validation		Testing	
	Acc_{Train} (%)	$Loss_{Train}$	Acc_{Val} (%)	$Loss_{Val}$	Acc_{Test} (%)	$Loss_{Test}$
50	99.04	0.0337	88.46	0.5054	87.5	0.3865
100	97.11	6.01×10^{-5}	97.11	0.1318	95.19	0.0981
200	99.36	0.0088	**98.1**	**0.0380**	**98.077**	**0.0696**
300	**100**	$\mathbf{1.19} \times 10^{-7}$	95.19	0.4544	91.35	0.7339

200 epochs is optimal for the 3D CNN training. Regarding the test process, the model trained with 200 epochs performs better on unseen data, reaching an accuracy of over 98%. Based on this analysis, we conclude that training our "3D FireClassNet" for 200 epochs is the most suitable choice.

5.3 Comparative study

In order to further demonstrate the effectiveness of the proposed 3D fire recognition model, a comparative analysis is conducted with the state-of-the-art models. Two types of fire recognition approaches, based on exploiting the spatio-temporal features, are used for this comparison: hand-crafted approaches (Dimitropoulos et al. 2012; Barmpoutis et al. 2013; Dimitropoulos et al. 2015; Torabian et al. 2021) and deep learning ones (Shin et al. 2018; Kim and Lee 2019; Vu et al. 2021; Nguyen et al. 2021). The results of these two comparisons are summarized in Tables 14 and 15, in the subsequent subsections. It is important to note that in these tables, the values presented in bold format indicate the best results, while the blank values marked with "NA" signify that the corresponding values are not reported in the related research.

5.3.1 Comparison with the existing video-based fire detection approaches

Table 14 presents the comparison findings between the developed 3D fire recognition model and the related works based on manual spatio-temporal features extraction. Each contribution is evaluated on a specific dataset. This is because there is no standard fire database, as previously mentioned. For an accurate comparison, it is suggested to assess our model on the datasets used by these approaches. In Dimitropoulos et al. (2012); Barmpoutis et al. (2013); Dimitropoulos et al. (2015), the "Firesense" dataset, containing 11 fire videos and 16 non-fire scenes (Grammalidis et al. 2017), is used for the experiments. Dimitropolous et al. focus on extracting spatial and temporal features using different descriptors. In Dimitropoulos et al. (2012), a motion descriptor is employed based on the spatio-temporal features, combined with a color model. These extracted features are merged with a new spatio-temporal consistency, in Barmpoutis et al. (2013). Afterwards, dynamic texture analysis is introduced in Dimitropoulos et al. (2015) to enhance the fire recognition performance. As shown, the False Positive Rate (*FPR*) is enhanced, in Dimitropoulos et al.'s works, from 13.8% in Dimitropoulos et al. (2012) to 0% in Dimitropoulos et al. (2015). Similarly, no false positives are identified by our model, when it is evaluated on the "Firesense" dataset. Besides, the accuracy value is 99.37% which is greater than the reached one 98.05% achieved by the model presented by

Table 14 Comparison of the presented 3D model performance with the existing hand-crafted methods

Approaches	Datasets	FPR (%)	R (%)	P (%)	A(%)
Hand-crafted methods					
Dimitropoulos et al. (2012)	Firesense Grammalidis et al. (2017)	13.80	96.31	NA	NA
Barmpoutis et al. (2013)	Firesense	0.08	96.18	NA	98.05
Dimitropoulos et al. (2015)	Firesense	0	99.17	NA	NA
Torabian et al. (2021)	Bilkent Cetin (2007)	0.03	99.52	NA	NA
DL-based method					
Proposed approach	Firesense	**0**	**98.21**	**100**	**99.37**
	Bilkent	0.0192	96.42	96.43	97.5
	Our dataset	0.01315	**99.46**	**99.46**	**99.23**

Table 15 Comparison of our 3D fire recognition model's performance with the related deep models

Approaches	Datasets	FPR (%)	R (%)	P (%)	A(%)
Shin et al. (2018)	Furg Steffens et al. (2015)	0.0395	NA	86.70	96.14
Kim and Lee (2019)	Mivia Foggia et al. (2015)	2.47	NA	NA	97.92
Vu et al. (2021)	Mivia	2.7	NA	NA	97.5
Nguyen et al. (2021)	Mivia	2.7	NA	92	93.3
	Furg	**0**	**100**	**100**	**100**
Proposed approach	Mivia	**0.0294**	**88.46**	**95.83**	**93.33**
	Our dataset	0.01315	99.46	99.46	99.23

Barmpoutis et al. (2013). It is demonstrated that, by giving sufficient data and computational power, our proposed 3D CNN architecture outperforms the traditional hand-crafted methods. These approaches, relying on manually extracting and analyzing features, might capture certain aspects of the data well but often struggle with high-dimensional nature of video data. In contrast, our proposed three-dimensional network automatically extracts and directly learns features from the data itself, enabling it to capture the spatio-temporal features of video sequences. Despite of the promising findings achieved by these approaches, they may not reach the higher performance as a 3D model. This is primarily because hand-crafted methods can miss out on important temporal information that is crucial for accurately recognizing and tracking fire in videos. The 3D CNN, on the other hand, processes the spatial and temporal dimensions simultaneously, offering a more comprehensive analysis of the video data.

Another dataset is employed by Torabian et al. in Torabian et al. (2021), to assess their fire recognition model, known as the "Bilkent" dataset (Cetin 2007). Their approach is based on extracting spatial and temporal features from each frame, which are then used in an SVM classifier to distinguish fires in video sequences. The difference in the False Positive Rate (*FPR*) between our model, evaluated on the "Bilkent" dataset and that of Torabian et al. does not exceed 0.0108%. This indicates their close effectiveness at reducing the false positives. The model, introduced in Torabian et al. (2021), also guarantees slightly better fire recognition performance than our 3D model, in terms of recall. The reached recall value of 96.42% can be explained by some misclassification, where there is a lot of fire scenes that are incorrectly recognized as non-fire fire videos. It's important to note that the "Bilkent" dataset comprises only 13 fire videos, which is insufficient for our "3D FireClassNet" training, needing two classes: fire and non-fire.

By analyzing this table, it is revealed that our 3D model outperforms the related hand-crafted methods in terms of False Positive Rate. This is particularly when testing on the datasets used in the related works. When the test performance is measured on our created dataset, better results are achieved, compared to reached findings when testing on the other dataset. This reflects the impact of the preprocessing phase, where the input data are augmented through different transformations. Our dataset includes diverse positions and orientations of fire object in video sequences. It is varied with challenging samples of fire and non-fire scenes. Therefore, our generated 3D model shows a greater ability to recognize fire and non-fire videos sequences, yielding to significant results. The outcomes highlight the advantages of deep learning models in recognition tasks compared to hand-crafted methods. These latter typically require more computational time and memory for feature extraction and recognition processes, since the whole recognition process is applied for each frame. In contrast, once a three-dimensional deep neural network is trained, the prediction process can be quickly accomplished. This is mainly because the 3D CNN simultaneously processes both spatial and temporal dimensions, providing a more accurate analysis of the video data.

5.3.2 Comparison with the related deep learning models based on spatio-temporal networks for fire recognition

Concerning the existing deep fire recognition models, that are based on spatio-temporal architectures, the proposed 3D model is compared with four related models (Shin et al. 2018; Kim and Lee 2019; Vu et al. 2021; Nguyen et al. 2021). Our model is evaluated using the datasets from these works, for a fair comparison. Table 15 presents the effectiveness comparison between our approach and the one proposed in Shin et al. (2018), using the "Furg" dataset (Steffens et al. 2015). It is composed by 17 fire videos and 6 non-fire scenes. As shown, our presented 3D model outperforms the Shin et al.'s model (Shin et al. 2018) in recognizing fire sequences, in terms of *FPR*, precision and accuracy. Indeed, the obtained *FPR* is lower than that of Shin et al. (2018). And the resulting values of precision and accuracy are greater than those achieved by Shin et al.'s model. This model is composed by two streams: the spatial stream is based on the VGG-16 network, and the temporal one is designed using a 3D CNN to capture the temporal information. Hence, it is demonstrated that the proposed "3D FireClassNet", based on a 3D convolutional neural network, provides better fire recognition performance, than that of Shin et al. (2018).

It can be also seen from this table that the "Mivia" dataset is utilized in Kim and Lee (2019); Vu et al. (2021); Nguyen et al. (2021). Using this dataset, containing 14 fire videos and 16 non-fire scenes, our presented model is more efficient than the three state-of-the-art models of Kim and Lee (2019); Vu et al. (2021); Nguyen et al. (2021). Their introduced networks are based on a CNN-LSTM architecture for the spatio-temporal analysis. In fact, our achieved precision is actually superior by 3.83% compared to the value belonging to the model of Nguyen et al. (2021). The False Positive Rate (*FPR*) is also improved with our "3D FireClassNet" by 2.6706%, which demonstrates the ability of our model to reduce the number of false positives. This reflects the best precision value of 95.83%, compared to the model's precision of 92%. It should be pointed that the results of our fire recognition model, evaluated on our constructed dataset, reaches the highest findings.

This advantage derives not only from the diversity of samples in our dataset, but also from the simplicity of the proposed 3D CNN structure. A 3D CNN is designed to simultaneously analyze spatial and temporal information

within a single network. It differs greatly from the CNN-LSTM and the two-stream networks, which require separate models to process spatial and temporal data. This leads to longer training time. These complex models need much more time to learn, as they involve the coordination of two separate networks. By using the presented "3D FireClassNet", this process is optimized by reducing overall training time while efficiently capturing the crucial temporal dynamics and spatial features required for accurate fire recognition. This efficiency makes the 3D CNN particularly well adapted to our diverse dataset, enabling robust performance without the extended training time needed by other methods.

To further assess our model's effectiveness, it is compared to other approaches based on spatio-temporal networks. Table 16 presents the comparative results. The precision and recall achieved with our model exceed those obtained by Zhikai Yang et al. who used a lightweight CNN and the Simple Recurrent Unit (SRU) Yang et al. (2020). Specifically, our model's precision is 99.46% compared to 97.4%. This low precision in Yang et al. (2020) shows that their model may require additional training or specific heuristics to capture more fire instances. The improved scores indicate that our 3D network structure yields more accurate fire recognition outcomes. This is thanks to the simplicity of the "3D FireClassNet" and its well-adjusted hyperparameters. More notably, when compared with studies that use the combination of CNN architecture with BLSTM and LSTM respectively (Agirman and Tasdemir 2022; Abhilash 2023), our model demonstrates a high overall performance with an accuracy of 99.23%. This proves that the simplified structure of the proposed "3D FireClassNet" along with the optimized hyperparameters for training, outperforms results obtained from the CNN-BLSTM and CNN-LSTM architectures. These ones, which require analyzing multiple consecutive video frames simultaneously, typically need longer training times and a larger number of parameters. Furthermore, our model exhibits superior performance with an impressive weighted harmonic mean F1 score of 99.46%, which is better than results from works employing ConvLSTM-based architectures with various modifications, like in (He et al. 2024; Masrur et al. 2024).

According to this comparison, our 3D CNN is the best performing architecture for fire recognition purposes. This is not only related to the varied and diversified samples that constitute our dataset, but also to the simple structure of a 3D CNN. Just one network is designed to analyze simultaneously spatial and temporal informations. It does not require a significant amount of time for training, unlike the CNN-LSTM or two-stream architectures, which need a lot of time to train two separate networks. Thanks to its uncomplicate structure, 3D CNN requires four dimensions: two spatial dimensions, one channel dimension and one temporal dimension. Therefore, the structure of the proposed "3D FireClassNet" with the adopted hyperparameters of the presented experimental protocol, along with a sufficient amount of data, contributes to the promising achieved results.

5.4 Comparison between "3D FireClassNet" and "FireClassNet" architectures

As it is mentioned, our proposed network "3D FireClassNet" is inspired by the presented "FireClassNet" of the work (Daoud et al. 2023), which is developed for fire image classification. Our idea is to expand the 2D spatial CNN into a spatio-temporal neural network using a 3D CNN, where spatio-temporal features are learned, and a time dimension is added to the input vector. This temporal dimension enhances the capability of the network to analyze video sequences and recognize the fire object. The main difference lies in the convolutional and max pooling layers: Conv2D with filter size of 2×2 and MaxPooling2D layers are applied in the 2D structure of "FireClassNet", whereas Conv3D with a filter kernel size of $3 \times 3 \times 3$ and MaxPooling3D layers are used in the 3D structure of "3D FireClassNet" network. The impact of employing Conv3D layers is significantly proved since they exploit spatio-temporal features rather than relying only on spatial features for fire image classification.

It is experimentally shown, from Table 17, that our model based on the 3D CNN performs better than the developed model for fire image classification approach. Actually, our "3D FireClassNet" is inspired by the 2D architecture "FireClassNet" presented in Daoud et al. (2023). During experiments, it is deduced that the training time of the 3D CNN

Table 16 Performance comparison with other related works

Approaches	Architectures	A (%)	P (%)	R (%)	F1(%)
Yang et al. (2020)	CNN + SRU	97.20	97.4	96.2	96.79
Agirman and Tasdemir (2022)	CNN + BLSTM	94.5	NA	NA	94.4
He et al. (2024)	Optimized ConvLSTM	92.17	NA	NA	89.37
Masrur et al. (2024)	ConvLSTM with attention mechanism	NA	98.40	61.93	76.02
Abhilash (2023)	CNN + LSTM	97.47	NA	NA	NA
Proposed	**3D CNN**	**99.23**	**99.46**	**99.46**	**99.46**

Table 17 Comparison between "FireClassNet" and "3D FireClassNet" networks

	"FireClassNet" (Daoud et al. 2023)	"3D FireClassNet"
Training time	1 d: 07 h: 39 min: 01s	01 h: 05 min: 50s
Number of parameters	6,630,978	4,935,074

d, day; h, hour; min, minute; s, second

network is reduced compared to the training time of "FireClassNet", as well as the prediction time. Predicting a video is faster than predicting each separate frame of a sequence. Hence, the huge time required for the prediction process in a single frame is reduced. This is because the model's input is a video where the spatial and temporal informations are simultaneously treated through a single-stream network. Add to that, concerning the number of parameters in each architecture, the "3D FireClassNet" has approximately 4 M parameters, compared to 6 M in the "FireClassNet" network. With these achievements, the main goals of this study have been accomplished, including the development of a simplified 3D CNN structure for fire video analysis. The derived model is characterized by having fewer parameters and reduced training and prediction times, making it suitable for deployment on small devices to recognize fires in video surveillance systems.

6 Conclusion

In this paper, a novel deep fire recognition approach in video surveillance scenes is introduced. It is principally based on a new 3D convolutional neural network "3D FireClassNet", which captures and learns spatial and temporal features using one stream network. This approach consists in preprocessing the input video data by applying various transformations. Subsequently, the proposed 3D CNN is trained on the varied video samples of the created dataset. An accurate 3D model, capable of recognizing fire in videos, is generated. The experimental outputs demonstrate the efficiency of the 3D CNN structure in learning the spatial information and the temporal interactions between adjacent frames. Comparative analysis also illustrates that the "3D FireClassNet" outperforms the related hand-crafted methods and spatio-temporal architectures in terms of false positive rate and precision. This proves the potential benefits of using 3D CNN in the fire recognition field, specifically to explore the spatial and temporal features simultaneously within a single network. Hence, the motion is captured more effectively than 2D CNN, yielding to improved performance and reduced training and prediction times.

Despite these achieved promising results, the developed 3D model has some limitations that should be considered in our follow-up work. Indeed, training the network with a large number of videos that cover diverse situations and contexts, can result in greater accuracy. To further enhance the model's performance, we propose enriching the dataset swith more fire and non-fire scenes, incorporating more challenging sequences. It is also planned to expand the fire recognition task to include smoke detection, where fires will be detected from their early stages starting with the appearance of smoke. These improvements for the upcoming research aim to create a more robust and efficient 3D CNN model for a comprehensive fire recognition system applicable to real-world scenarios.

Acknowledgements The authors would like to acknowledge Deanship of Graduate Studies and Scientific Research, Taif University for funding this work.

Author contributions We confirm that the manuscript has been read and approved by all named authors and that there are no other persons who satisfied the criteria for authorship. All authors contributed equally to this work.

Data availability The codes and data generated and analyzed during the current study are available from the corresponding author on reasonable request.

Declarations

Conflict of interest All authors declare that they have no conflict of interest.

References

Abhilash S (2023) Real time fire detection using deep convolutional neural networks and long-short term memory in video surveillance. Int J Res Appl Sci Eng Technol 11:1685–1693. https://doi.org/10.22214/ijraset.2023.53888

Agirman AK, Tasdemir K (2022) BLSTM based night-time wildfire detection from video. PLoS ONE 17(6):0269161

Avazov K, Mukhiddinov M, Makhmudov F, Cho YI (2022) Fire detection method in smart city environments using a deep-learning-based approach. Electronics. https://doi.org/10.3390/electronics11010073

Barmpoutis P, Dimitropoulos K, Grammalidis N (2013) Real time video fire detection using spatio-temporal consistency energy. In: 2013 10th IEEE international conference on advanced video and signal based surveillance, pp 365–370. https://doi.org/10.1109/AVSS.2013.6636667

Bhat O, Khan DA (2022) Evaluation of deep learning model for human activity recognition. Evol Syst 13(1):159–168

Cazzolato MT, Avalhais L, Chino D, Ramos JS, Souza JA, Rodrigues JF Jr, Traina A (2017) Fismo: a compilation of datasets from emergency situations for fire and smoke analysis. In: Brazilian symposium on databases-SBBD, pp 213–223. Available at http://sbbd.org.br/2017/wp-content/uploads/sites/3/2017/10/proceedings-satellite-events-sbbd-2017.pdf

Celik T (2010) Fast and efficient method for fire detection using image processing. ETRI J 32(6):881–890. https://doi.org/10.4218/etrij.10.0109.0695

Center Philanthropy (2022) International wildfires. Available at https://disasterphilanthropy.org/disasters/2022-international-wildfires/

Cetin E (2007) Computer vision based fire detection software. Available at http://signal.ee.bilkent.edu.tr/VisiFire/

Çetin AE, Dimitropoulos K, Gouverneur B, Grammalidis N, Günay O, Habiboğlu YH, Töreyin BU, Verstockt S (2013) Video fire detection-review. Digit Signal Process 23(6):1827–1843. https://doi.org/10.1016/j.dsp.2013.07.003

Chandra P, Das R (2023a) A hybrid RSA-IPA optimizer for designing an artificial neural network to study the Jeffery–Hamel blood flow with copper nanoparticles: application to stenotic tapering artery. Results Eng 20:101542

Chandra P, Das R (2023b) Finite-element-based machine-learning algorithm for studying gyrotactic-nanofluid flow via stretching surface. Int J Numer Methods Fluids 95(12):1888–1912

Chandra P, Das R (2024) A hybrid machine learning algorithm for studying magnetized nanofluid flow containing gyrotactic microorganisms via a vertically inclined stretching surface. Int J Numer Methods Biomed Eng 40(1):3780

Chen T-H, Wu P-H, Chiou Y-C (2004) An early fire-detection method based on image processing. In: 2004 international conference on image processing, 2004. ICIP'04, vol 3. IEEE, pp 1707–1710. https://doi.org/10.1109/ICIP.2004.1421401

Daoud Z, Ben Hamida A, Ben Amar C (2023) FireClassNet: a deep convolutional neural network approach for PJF fire images classification. Neural Comput Appl. https://doi.org/10.1007/s00521-023-08750-3

Dimitropoulos K, Tsalakanidou F, Grammalidis N (2012) Flame detection for video-based early fire warning systems and 3D visualization of fire propagation. In: 13th IASTED international conference on computer graphics and imaging (CGIM 2012), Crete, Greece

Dimitropoulos K, Barmpoutis P, Grammalidis N (2015) Spatio-temporal flame modeling and dynamic texture analysis for automatic video-based fire detection. IEEE Trans Circ Syst Video Technol 25(2):339–351. https://doi.org/10.1109/TCSVT.2014.2339592

Eberle C, Higuera Roa O (2022) Technical report: Mediterranean wildfires

Foggia P, Saggese A, Vento M (2015) Real-time fire detection for video-surveillance applications using a combination of experts based on color, shape, and motion. IEEE Trans Circ Syst Video Technol 25(9):1545–1556. https://doi.org/10.1109/TCSVT.2015.2392531

Gayathiri B, Brindha P, Karthika I, Saranya E, Rajeshkumar G, Rajesh Kanna P (2023) Machine learning based crop suitability prediction and fertiliser recommendation system. In: 2023 4th international conference on electronics and sustainable communication systems (ICESC), pp 1023–1028. https://doi.org/10.1109/ICESC57686.2023.10193542

Glorot X, Bengio Y (2010) Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the thirteenth international conference on artificial intelligence and statistics. JMLR workshop and conference proceedings, pp 249–256

Grammalidis N, Dimitropoulos K, Cetin E (2017) FIRESENSE database of videos for flame and smoke detection. Zenodo. Available at https://zenodo.org/record/836749/files/fire_videos.1406.zip?download. https://doi.org/10.5281/zenodo.836749

Harsha R, Karthika S, Mohana PD, Rajesh KP, Rajeshkumar G, Mythili E (2023) Folla disease detection using in-depth learning. In: 2023 2nd international conference on applied artificial intelligence and computing (ICAAIC). IEEE, pp 547–552

He K, Zhang X, Ren S, Sun J (2015) Delving deep into rectifiers: surpassing human-level performance on ImageNet classification. In: Proceedings of the IEEE international conference on computer vision, pp 1026–1034

He Z, Fan G, Li Z, Li S, Gao L, Li X, Zeng Z-C (2024) Deep learning modeling of human activity affected wildfire risk by incorporating structural features: a case study in eastern China. Ecol Indic 160:111946. https://doi.org/10.1016/j.ecolind.2024.111946

Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Kanna PR, Santhi P (2021) Unified deep learning approach for efficient intrusion detection system using integrated spatial–temporal features. Knowl Based Syst 226:107132

Kanna PR, Santhi P (2022) Hybrid intrusion detection using MapReduce based black widow optimized convolutional long short-term memory neural networks. Expert Syst Appl 194:116545

Karthika S, Priyanka T, Indirapriyadharshini J, Sadesh S et al (2023) Prediction of weather forecasting with long short-term memory using deep learning. In: 2023 4th international conference on smart electronics and communication (ICOSEC), pp 1161–1168. https://doi.org/10.1109/ICOSEC58147.2023.10276273

Khalil A, Rahman SU, Alam F, Ahmad I, Khalil I (2021) Fire detection using multi color space and background modeling. Fire Technol 57:1221–1239. https://doi.org/10.1007/s10694-020-01030-9

Kim B, Lee J (2019) A video-based fire detection using deep learning models. Appl Sci 9:2862

Lakshmi M, Das R, Manohar B (2024) A new Covid-19 classification approach based on Bayesian optimization SVM kernel using chest X-ray datasets. Evol Syst. https://doi.org/10.1007/s12530-024-09575-8

Manohar B, Das R (2022a) Artificial neural networks for the prediction of monkeypox outbreak. Trop Med Infect Dis 7(12):424. https://doi.org/10.3390/tropicalmed7120424

Manohar B, Das R (2023b) Artificial neural networks for prediction of Covid-19 in India by using backpropagation. Expert Syst 40(5):13105. https://doi.org/10.1111/exsy.13105

Manohar B, Das R (2023) Comparison of hybrid artificial neural networks with GA, PSO, and RSA in predicting Covid-19 cases: a case study of India. In: Multi-disciplinary applications of fog computing: responsiveness in real-time. IGI Global, New York, pp 207–244. https://doi.org/10.4018/978-1-6684-4466-5.ch011

Masrur A, Yu M, Yu M (2024). Deciphering wildfire spread dynamics: attention-based spatiotemporal models using CONV-LSTM networks. https://doi.org/10.2139/ssrn.4695951

Mehta R, Singh KK (2023) An efficient ear recognition technique based on deep ensemble learning approach. Evol Syst 2023:1–17

Muhammad K, Khan S, Elhoseny M, Hassan Ahmed S, Wook Baik S (2019) Efficient fire detection for uncertain surveillance environment. IEEE Trans Ind Inform 15(5):3113–3122. https://doi.org/10.1109/TII.2019.2897594

Nguyen MD, Vu HN, Pham DC, Choi B, Ro S (2021) Multistage real-time fire detection using convolutional neural networks and long short-term memory networks. IEEE Access 9:146667–146679. https://doi.org/10.1109/ACCESS.2021.3122346

Nithya T, Dhivya P, Sangeethaa S, Kanna PR (2024) TB-MFCC multifuse feature for emergency vehicle sound classification using multistacked CNN-attention BILSTM. Biomed Signal Process Control 88:105688

Phillips Iii W, Shah M, Vitoria Lobo N (2002) Flame recognition in video. Pattern Recogn Lett 23(1–3):319–327. https://doi.org/10.1016/S0167-8655(01)00135-0. Available at https://www.crcv.ucf.edu/data/fire.php

Rasool Abdali AM, Ghani RF (2019) Robust real-time fire detector using CNN and LSTM. In: 2019 IEEE student conference on research and development (SCOReD), pp 204–207. https://doi.org/10.1109/SCORED.2019.8896246

Shi X, Chen Z, Wang H, Yeung D-Y, Wong W-K, Woo W-C (2015) Convolutional LSTM network: a machine learning approach for precipitation nowcasting. Adv Neural Inform Process Syst 28:1

Shin J, Park H, Paik J (2018) Fire recognition using spatio-temporal two-stream convolutional neural network with fully connected layer-fusion. In: 2018 IEEE 8th international conference on consumer electronics-Berlin (ICCE-Berlin), pp 1–3. https://doi.org/10.1109/ICCE-Berlin.2018.8576218

Simonyan K, Zisserman A (2014) Two-stream convolutional networks for action recognition in videos. Adv Neural Inform Process Syst 27:1

Statista (2022) Area burned by wildfires in Tunisia from 2009 to 2022. Available at https://www.statista.com/statistics/1322257/area-burned-by-wildfire-in-tunisia/#statisticContainer

Steffens CR, Rodrigues RN, Costa Botelho S (2015) An unconstrained dataset for non-stationary video based fire detection. In: 2015 12th Latin American robotics symposium and 2015 3rd Brazilian symposium on robotics (LARS-SBR), pp 25–30. https://doi.org/10.1109/LARS-SBR.2015.10. Available at https://github.com/steffensbola/furg-fire-dataset

Torabian M, Pourghassem H, Mahdavi-Nasab H (2021) Fire detection based on fractal analysis and spatio-temporal features. Fire Technol 57(5):2583–2614

Tran D, Bourdev L, Fergus R, Torresani L, Paluri M (2015) Learning spatiotemporal features with 3D convolutional networks. In: Proceedings of the IEEE international conference on computer vision, pp 4489–4497

Verlekar TT, Bernardino A (2020) Video based fire detection using Xception and CONV-LSTM. In: Advances in visual computing: 15th international symposium, ISVC 2020, San Diego, CA, USA, October 5–7, 2020, proceedings, part II, vol 15. Springer, London, pp 277–285. https://doi.org/10.1007/978-3-030-64559-5_21

Vrskova R, Hudec R, Kamencay P, Sykora P (2022) A new approach for abnormal human activities recognition based on CONV-LSTM architecture. Sensors 22(8):2946. https://doi.org/10.3390/s22082946

Vu HN, Tran AD, Nguyen MD, Choi B, Ro S (2021) Investigation of deep learning method for fire detection from videos. In: 2021 international conference on information and communication technology convergence (ICTC), pp 593–595. https://doi.org/10.1109/ICTC52510.2021.9621042

Wahyono Harjoko A, Dharmawan A, Adhinata FD, Kosala G, Jo K-H (2022) Real-time forest fire detection framework based on artificial intelligence using color probability model and motion feature analysis. Fire 5(1):23. https://doi.org/10.3390/fire5010023

Yang Z, Bu L, Wang T, Yuan P, Jineng O (2020) Indoor video flame detection based on lightweight convolutional neural network. Pattern Recogn Image Anal 30:551–564. https://doi.org/10.1134/S1054661820030293