# An Efficient Deep Learning Algorithm for Fire and Smoke Detection with Limited Data

Abdulaziz NAMOZOV, Young Im CHO
*Department of Computer Engineering, Gachon University*
*Seoul, 13120 – Korea Republic*
*yicho@gachon.ac.kr*

*Abstract*—**Detecting smoke and fire from visual scenes is a demanding task, due to the high variance of the color and texture. A number of smoke and fire image classification approaches have been proposed to overcome this problem; however, most of them rely on either rule-based methods or on handcrafted features. We propose a novel deep convolutional neural network algorithm to achieve high-accuracy fire and smoke image detection. Instead of using traditional rectified linear units or tangent functions, we use adaptive piecewise linear units in the hidden layers of the network. We also have created a new small dataset of fire and smoke images to train and evaluate our model. To solve the overfitting problem caused by training the network on a limited dataset, we improve the number of available training images using traditional data augmentation techniques and generative adversarial networks. Experimental results show that the proposed approach achieves high accuracy and a high detection rate, as well as a very low rate of false alarms.**

*Index Terms*—**smoke detectors, neural networks, image classification, image recognition, image generation.**

## I. Introduction

Detecting fire or smoke at an early stage is a crucial to facilitate intervention in time to avoid large-scale damage. Several methods and tools have been using to recognize fire or smoke in visual scenes. Most of the traditional fire and smoke detection methods use sensor-based tools, which are supposed to detect the presence of the smoke or flame. The main disadvantage of such sensors is that they can generally only recognize fire or flame near the places where they are installed, and this limits their effectiveness in large covered areas. Moreover, flame or smoke detection sensors cannot provide enough information about the direction, initial location, or size of the fire.

Various video and image-based algorithms have been suggested for overcoming the imperfections of the above-mentioned techniques. These methods, outlined below, use images or videos captured by cameras, which can be installed in both indoor and outdoor environments to detect fire and smoke. Chen et al. [1] suggested a smoke detection method based on video processing. A chromatically-based static decision rule and diffusion-based dynamic decision rule is used to draw conclusions about whether each pixel does or does not represent smoke. Another method, provided by Toreyin et al. [2], uses motion and color clue features combined with edge blurring and flicker features, to detect fire and flame. A video-based optical flow method was developed by Mueller et al. [3], essentially an optimal mass transport model using the features of the flow direction and magnitude to differentiate between fire and non-fire motion. Computer vision based measurement method which uses GIS-based augmented reality to measure the dimensions of the flame was proposed by Bugarovich et al. [4]. Yet other approaches have been suggested that create a rule-based algorithm using the color, shape, and temporal aspects of fire and the smoke in images [5].

The main purpose of the above-mentioned methods is to create a rule-based algorithm that relies on handcrafted features, or specific expert knowledge. The main features of smoke and fire can be very diverse, depending on many factors including formation, texture, and color, making it a complicated task for those algorithms to achieve high accuracy in detecting fire and smoke from natural images.

Another approach is using deep learning algorithms to detect and extract important features of color and smoke from images and videos. Despite the fact that deep learning, particularly convolutional neural networks (CNN), achieves excellent results in solving visual recognition problems, only a few studies have been undertaken using these methods in smoke or fire detection. For instance, Zhang et al. [6] proposed two joined deep CNNs to detect fire in forest images. In this approach, firstly, the whole image is tested by a global image classifier; then if a fire is detected in any region, another classifier is used to identify the exact location of the fire in that image. A deep CNN model for smoke detection was suggested by Tao et al. [8], inspired by AlexNet [7]. While those approaches use existing deep learning methods for fire or image detection, Yin et al. [9] developed a deep normalization and CNN model for image smoke detection.

The goal of most of the existing approaches is detecting either smoke or fire from images, but as explained, they suffer from a variety of limitations. To solve the problem of these limitations, in this paper, we propose a novel deep CNN model for fire and smoke image classification. This method is capable of classifying both smoke and fire images at the same time, and offers many advantages and exhibits better performance than other existing visual recognition CNN models for the recognition of fire and smoke in images.

Another contribution of our work is a new dataset which consists of 2440 fire and smoke images. As far we know there is no any public dataset for smoke or fire image detection, and thus, we create a dataset of fire and smoke images that can be used to train our network and test our method. As CNN models require a huge quantity of images to train effectively, we use various data enhancement techniques to create more training examples from the

Digital Object Identifier 10.4316/AECE.2018.04015

limited number of original images of our data benchmark. We use a combination of traditional image transformation methods and generative adversarial networks (GANs) [10] for generating new training images, which boosts the performance of our network as well as to solving the problem of overfitting.

The rest of the paper is organized as follows. Section II gives a brief review of deep CNN and different data augmentation techniques. The architecture of our novel network for fire and smoke image classification, as well as data enhancement techniques that we use, are explained in Section III. Experimental results and a comparison of the performance of our model with different state-of-the-art models detailed in Section IV. Section V concludes the paper.

## II. Related Work

This section provides a review of previous research on deep CNN for pattern recognition, and data augmentation techniques to improve the performance of the classification algorithms

### 1) *Convolutional Neural Networks*

The development of CNNs brought notable improvements in the performance of many computer vision tasks, in particular, visual recognition and image classification problems. LeCun et al. [11] achieved one of the very first successful results in this field by proposing a CNN algorithm called LeNet-5 which was applied to recognize hand-written characters. Since then, researchers have found that the classification performance of a CNN can be improved by increasing its depth. However, deeper networks also demand more computational capacity and energy consumption. Recently, the availability of large-scale datasets and the development of very powerful GPUs have allowed researchers to make CNNs very deep. For instance, a deep CNN network called AlexNet, proposed by Krizhevsky et al. [7], achieved excellent performance (top-1 and top-5 error rates of 37.7% and 17.0%) in the 2012 ImageNet Large Scale Visual Recognition Challenge. There was no clear understanding of why deep CNNs show very good classification performance until the visualization technique proposed by Zeiler et al. [12] provided an intuitive understanding of the functions of intermediate feature layers. Additionally, some other algorithms based on deep CNNs have also demonstrated very good performance in image classification, such as [13-17].

### 2) *Data Augmentation*

The reason behind the success of deep CNNs in visual recognition problems is not only the development of better network architectures, or the presence of powerful computational environments, but the availability of a vast quantity of image data. Most of the successful deep CNNs achieved good results when they trained with thousands of high-resolution images. However, when it comes to some specific tasks, such as fire or smoke detection problems, such large datasets are not available for training. The main problem of neural network models trained with limited data is overfitting, which means they cannot generalize data well from the validation and test set. Use of data augmentation techniques for creating new training samples is considered one of the simplest way of reducing the overfitting problem in neural networks [12].

The most common type of data enhancement technique is enlarging the image dataset using label-preserving transformations [18]. Performing other geometric and color augmentations, such as cropping part of the image, rotation, scaling, flipping, shearing, or changing the color space of original images can also be used to improve a number of training examples.

Another way of creating new training image samples is using generative models. GANs [10] is an algorithm for estimating generative models that use the min-max strategy, where a generative model G generates similar samples from the original data distribution, and a discriminative model D is trained to estimate the probability that a sample came from training data rather than G.

A method for image-to-image translation using cycle-consistent adversarial networks (CycleGAN), proposed by Zhu et al. [19], can be used for transferring images in one setting to another. For instance, images of different seasons, or different weather conditions can be generated by using only data collected on one season/weather conditions. This image generation technique for data augmentation improves the diversity of the image data.

## III. Proposed Approach

### 1) *Image Dataset*

Deep learning methods for image recognition tasks are often based on learning millions of parameters. Additionally, this process requires the collection of a huge number of images. To the best of our knowledge, there is not any public data benchmark available which consists of smoke and fire images. Therefore, we created a dataset of such images, called SetA, which consists of 2440 high-resolution natural images as listed in Table I. Some sample images from this dataset can be seen in Fig. 1.

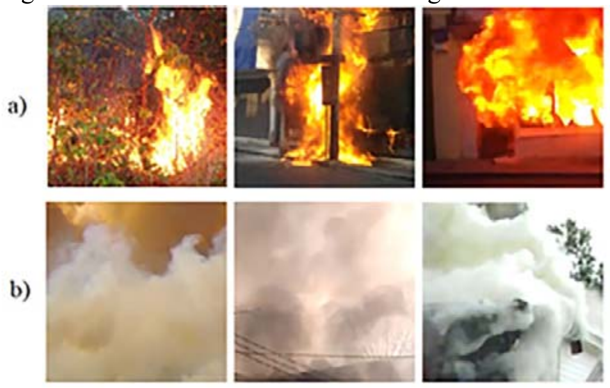

Figure 1. Example images from the *SetA* image dataset: a) fire images; b) smoke images.

### 2) *Data Augmentation*

Despite having a deep and robust CNN algorithm, high recognition cannot be achieved unless a huge number of training images are provided. This is because of the overfitting problem, where a network trained with a small number of images cannot generalize well on new, unseen test data. To solve this problem, we increased the number of training images using some data augmentation techniques.

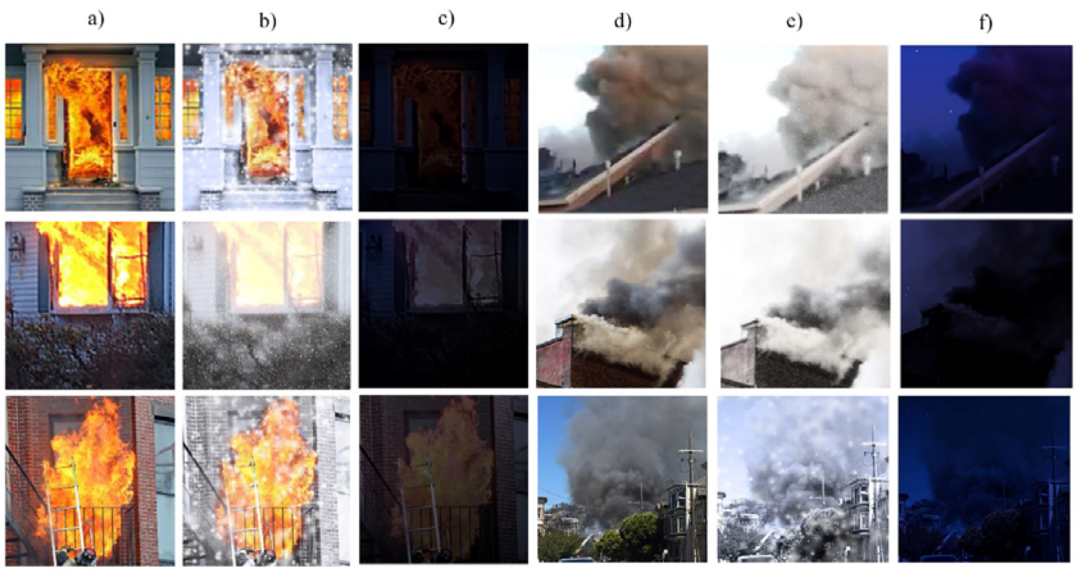

Figure 2. Some example images generated using GANs: a, d) original fire and smoke images; b, e) new winter images generated using GANs; c, f) new night images generated by GANs

TABLE I. ORIGINAL AND AUGMENTED IMAGE DATASETS FOR FIRE AND SMOKE IMAGE CLASSIFICATION SYSTEM

| Dataset | Number of fire images | Number of smoke images | Type |
|---|---|---|---|
| SetA | 1220 | 1220 | Original |
| SetB | 2250 | 2250 | Augmented |
| SetC | 1282 | 1248 | Augmented |
| SetD | 4282 | 4248 | Augmented |

### a) Traditional Image Transformations

Generating new images using traditional image transformations includes making color and geometric augmentations such as changing the color palette, or cropping and translating the image by some degrees. Considering the key role that color features play in recognition problems for smoke and fire images, we use only geometric transformation. In this work, for every input image in the training set, we created other new 4 images using horizontal and vertical flip and rotation by 90 and 180 degrees. We generate SetB, consists of the new images that were created using geometric transformations as listed in Table I.

### b) Generative Adversarial Networks

Despite the fact that increasing the number of training images using standard data augmentation helps a network to learn features more effectively, this technique generates only limited alternative data. In order to improve not only the number of images, but also the diversity of those images that we use for training, we use the Unpaired Image to Image Translation using Cycle-Consistent Adversarial Networks [19] and the Image-to-image Translation with Conditional Adversarial Networks [20] methods for data augmentation. The idea of using these image style transfer techniques is to create new images so our network can learn more features of fire and smoke as these appear in different seasons, styles, and light conditions.

Firstly, for each fire and smoke image of our original dataset, we generate new images using the season transfer technique of the CycleGAN algorithm [19]. There are 747 fire images and 695 smoke images in our dataset, which were taken in the summertime. To learn to mimic the style of winter photos, we used some seasonal images of Yosemite (both winter and summer) downloaded from Flickr as explained in [19], and other images that were taken on snowy days. We follow the full training and image generation process outlined in [19]; post-training, we use only summer photos from our dataset to create winter styles for each image. This method helps us to create 747 new fire images and 695 new smoke images that resemble winter photos.

Secondly, for each daytime photo of our fire and smoke image dataset, we create night-time images using the idea proposed in [20]. There are 532 day-time fire images and 498 day-time smoke images in our original dataset, and so 1030 new fire and smoke images are generated using this method. Then, we create a new dataset called SetC which includes images created by both GAN methods mentioned above, as listed in Table I. Some examples of the new images generated by GANs can be seen in Fig. 2.

### 3) Network Architecture

We have created a novel CNN for fire and smoke image classification using some of the ideas fundamental to VGG-Net [16], which showed excellent results in the classification task of the 2014 ImageNet Challenge. Considering the fact that that we only solve a three-class problem, we create a network that is not deep as the original VGG-net; it consists of 12 layers, as shown in Fig. 3. The network has six convolutional and three pooling layers for feature extraction, and three fully-connected layers, in the end, for classification.

The main modification we make to the VGG-Nets approach in our CNN model is using the adaptive piecewise linear activation (APL) function [21], instead of using traditional ReLU functions in convolutional layers. This contribution improves the performance of the network, despite having some additional features which increase the training time (the detailed experimental results in are shown Section IV). Another difference between our deep CNN and the original VGG-net is we use a different number of filters in convolutional and pooling layers.

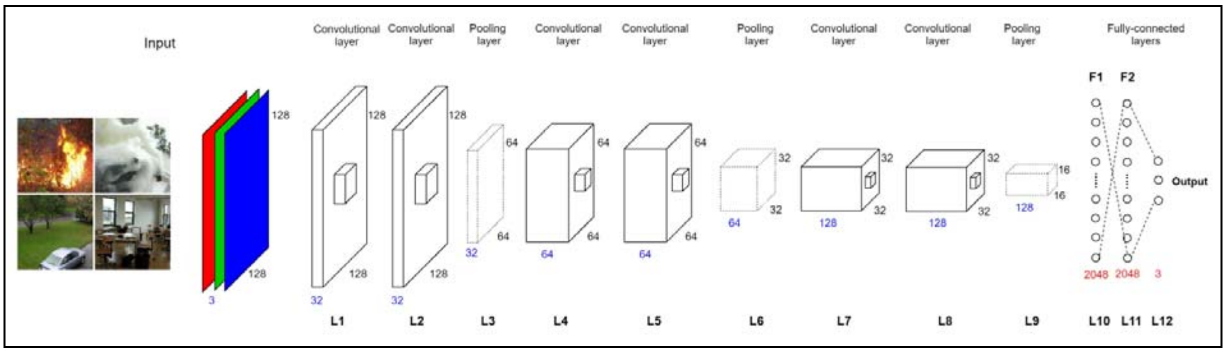

Figure 3. The architecture of the deep CNN for fire and smoke image classification. The network consists of twelve layers: six convolutional layers (cuboids with solid lines), three max-pooling layers (cuboids with dotted lines) and three fully connected layers at the end. Blue numbers under cuboids denote the number of filters, black numbers denote the widths and heights of the feature maps and red numbers denote the number of neurons in the fully-connected layers.

### *a) Convolutional Layers*

We use small-size 3 x 3 convolutional filters for all convolutional layers. Each of first two convolutional layers has 32 feature maps, and after each pooling layer, we double the number of filters. Therefore, the third and fourth convolutional layers include 64 feature maps, whereas there are 128 feature maps in each last convolutional layer.

### *b) Adaptive Piecewise Linear Units*

Traditionally, rectified linear units (ReLUs) have been used in most famous CNN models as they tend to be much faster than tangent and sigmoid functions [12]. However, we use APL units, proposed by Agostinelli *et al.* [21], instead of using traditional ReLUs in our CNN model. As we show in Section IV, this approach improves the classification performance of our network.

The activation function $h_i(x)$ of an APL unit $i$ is a sum of hinge-shaped functions,

$$h_i = \max(0, x) + \sum_{s=1}^{S} a_i^s \max(0, -x + b_i^s) \qquad (1)$$

The result of Eq. (1) is piecewise linear activation function which computes the output of each neuron in feature extraction layers. Here, *S* denotes the number of hinges, and it is set as a hyperparameter. Variables $a_i^s$ and $b_i^s$ for $i \in 1,...,S$ are learned using gradient descent during training. The $a_i^s$ variables control the slopes of the linear segment, while the $b_i^s$ variables determine the location of images.

Using APL units requires the additional number of parameters to be learned; this is computed by *2SM*, where *M* denotes the total number of hidden units in the network. As the number of hidden layers (*M*) in our network is predefined, the proper value for *S* which defines the complexity of the activation function is determined during the validation process and it is explained in Section IV, B.

### *c) Pooling Layers*

Feature maps in convolutional layers are used to extract features. The role of pooling layers in CNNs is to attain special invariance by reducing the resolution of those feature maps. This can help to decrease the computational time, as well as to address the risk of the overfitting problem. Traditionally, pooling layers are connected to convolutional layers.

Average pooling or max pooling is most commonly used in deep learning models. The former uses the average activation value over a pooling region, whereas the latter selects the maximum activation value. The max pooling technique is used in our network. We apply the same method for all three pooling layers over a 2 x 2 pixel window with stride 2.

### *d) Fully-connected Layers*

After convolutional and pooling layers, we add three fully-connected layers to perform the classification. Generally, fully-connected layers include most of the learnable parameters. In our case, the first two fully-connected layers have 2048 channels each, and the third layer (which is the output layer of the network) contains three neurons and performs 3-way fire, smoke, and background image classification. To prevent overfitting, we use the dropout technique in the first two fully-connected layers. We use the softmax function, which computes the probability for each class in the third fully-connected layer.

Let the output of the second fully-connected layer be $F^2$, and probabilities for three classes be $\bar{y} = [\bar{y}_1, \bar{y}_2, \bar{y}_3]^T$ . The probability $\bar{y}_j$ for any class *j* is computed by the softmax function:

$$\bar{y}_j = \frac{\exp(F_j^2)}{\sum_{i=1}^{3} \exp(F_j^2)}, \quad j = 1,2,3 \qquad (2)$$

## 4) *Network Training*

The training procedure of our network generally follows the original VGG-net [16] procedure. All training-relevant hyper-parameters are set as given in Table II.

TABLE II. HYPER-PARAMETERS FOR TRAINING

| Hyper-parameters | Mini-batch size | Momentum | Dropout ratio | Initial learning rate | Learning rate decay coefficient |
|---|---|---|---|---|---|
| **Value** | 64 | 0.9 | 0.5 | 0.01 | 0.005 |

Weight initialization in neural networks is very important since the better the initialization, the better the network can learn. We initialize the weights using Xavier initialization [22], and the network is trained using stochastic gradient descent.

TABLE III. IMAGE DATASETS FOR TRAINING, TESTING, AND VALIDATION

| Dataset | Number of fire images | Number of smoke images | Number of background images | Total number of images | Purpose |
|---|---|---|---|---|---|
| Set1 | 750 | 750 | 825 | 2325 | Training |
| Set2 | 3000 | 3000 | 3000 | 9000 | Training |
| Set3 | 1282 | 1248 | 1300 | 3830 | Training |
| Set4 | 4282 | 4248 | 4300 | 12830 | Training |
| Set5 | 220 | 220 | 450 | 890 | Validation |
| Set6 | 250 | 250 | 500 | 1000 | Test |

## IV. EXPERIMENTAL RESULTS AND ANALYSIS

All experiments are performed using the Keras software package on the Ubuntu 16.04 operating system, running on a PC with Intel(R) Core(TM) i7-7700HQ CPU 2.80 GHz with an Nvidia GTX 1050 Ti GPU. For comparative purposes, we also implement some of the other leading deep CNNs using Tensorflow software package [24].

To carry out our experiments, we create six image datasets using our original images in *SetA* and the augmented images in *SetB* and *SetC* (Table I). We name these sets *Set1, Set2, Set3, Set4, Set5*, and *Set6*, as given in Table III. The first four datasets in the table are used for training, whereas *Set5* and *Set6* are used for validation and testing, respectively.

The validation set consists of 890 images, with an equal number of smoke and fire images (220), and 450 images with no smoke or fire which we call background images. *Set6*, used for testing and evaluating our results, has 1000 images in total, once again with an equal (250) number of smoke and fire images, and 500 background images. Both validation and test sets are created by using smoke and fire images from our original dataset (*SetA*; refer to Table I in Section III-A) and adding background images which are taken from various publicly available datasets.

In *Set2*, which consists of 9000 images, there are 3000 smoke, fire, and background images. Smoke and fire images are taken from *SetB* which consists of newly generated images resulting from the application of traditional image translation techniques. We also add smoke and fire images from our original dataset, and 3000 background images, to construct a new, larger, balanced training set.

Another large training set is created using smoke and fire images generated by GANs. We call this dataset *Set3*, and it has 1248 smoke images and 1282 fire images generated by using generative nets, and 1300 background images.

The largest dataset for the fire and smoke classification system is created by combining images from our original dataset and images that are generated by using traditional image transformations and GANs. We call this dataset *Set4I*, and it has 12830 images comprised of 4282 fire images, 4248 smoke images, and 4300 background images.

### *1) Evaluation Methods*

In order to compare the success of using data augmentation methods to improve the performance of the network, the role of some hyper-parameters, and how our network rates against some other deep CNN models, we use evaluation methods from [23], defined as follows:

$$DR = \frac{P_p}{Q_p} \times 100\%, \tag{3}$$

$$FAR = \frac{N_p}{Q_n} \times 100\%, \tag{4}$$

$$AR = \frac{P_p + N_n}{Q_n + Q_p} \times 100\%, \tag{5}$$

where DR, FAR, and AR are Detection Rate, False Alarm Rate, and Accuracy Rate, respectively. Moreover, $Q_p$ denotes the number of positive samples, which is the total number of smoke and fire images in our test data, and $Q_n$ denotes the number of negative examples (background images); $P_p$ and $N_p$ are the number of correctly classified positive examples and a number of falsely classified negative examples respectively. Our goal is to achieve a high accuracy rate (AR) and high true positive rate (TPR) as well as a low false alarm rate (FAR) simultaneously.

### *2) Effects of the APL unit hyperparameter*

To show the role of changing the APL unit hyper-parameter, S, that controls the complexity of the activation function, we list the evaluation results of our deep CNN with different S values in Table III. We train the network using Set1 with varying values of S and evaluate the results using the validation set, Set5. Results show that when we set S = 5, we achieve the best AR and FAR, with the second-best performance for DR.

TABLE IV. EVALUATION RESULTS OF THE NETWORK WITH DIFFERENT HYPER-PARAMETER VALUES IN APL UNITS

| Value of S | AR (%) | DR (%) | FAR (%) | Additional learnable parameters |
|---|---|---|---|---|
| *S = 1* | 94.05 | 92.58 | 0.79 | 11 |
| *S = 3* | 94.33 | **93.15** | 0.68 | 33 |
| *S = 5* | **94.85** | 93.12 | **0.63** | 55 |
| *S = 7* | 93.47 | 92.07 | 0.82 | 77 |

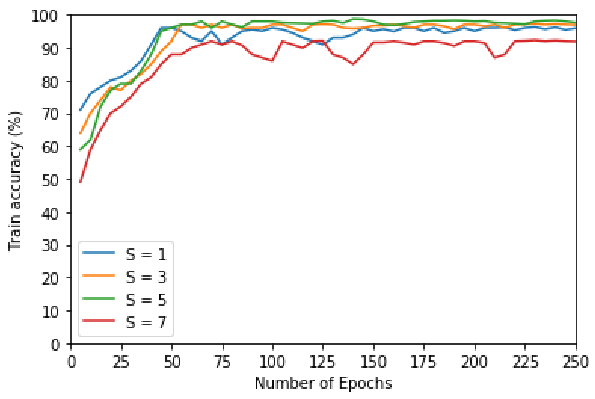

Figure 4. Training curves of a deep CNN with different values of *S*, in APL units.

Moreover, changing the value of S results in a change in the number of additional learnable parameters. Inevitably, this has an influence on the training process as illustrated in Fig. 3. When we set S = 1, training accuracy reaches approximately 96% after about 35 epochs and remains stable. As for S = 3, and S = 5, despite training for more epochs to achieve the highest training accuracy (around 98%), these values are more suitable than others because of the advantage offered in the performance of evaluation results as given in Table IV. When S =7, training accuracy does not surpass 93%, the worst case for our problem.

### 3) *Experiments with different activation functions*

To demonstrate the advantages of using APL units over traditional ReLUs and tangent functions, we compare the results of our deep CNN network model with APL units in hidden layers versus ReLU functions and tangent functions. We only change the activation functions of our model to ReLU and tangent functions; other hyperparameters remain unchanged. We train all networks using Set1 and evaluate using Set6. Evaluation results, listed in Table V, show that our network performs better with APL units in hidden layers than with ReLUs and tangent functions. Specifically, we can achieve the highest AR of 94.85% and DR of 93.12%, as well as the lowest FAR of 0.63%, when we use APL units in the hidden layers of our deep CNN model.

TABLE V. EVALUATION OF PERFORMANCE OF OUR DEEP CNN WITH DIFFERENT ACTIVATION FUNCTIONS

| **Activation functions** | **AR (%)** | **DR (%)** | **FAR (%)** |
|---|---|---|---|
| ReLU | 93.51 | 90.68 | 0.91 |
| Tangent | 91.75 | 89.87 | 1.21 |
| APL units | **94.85** | **93.12** | **0.63** |

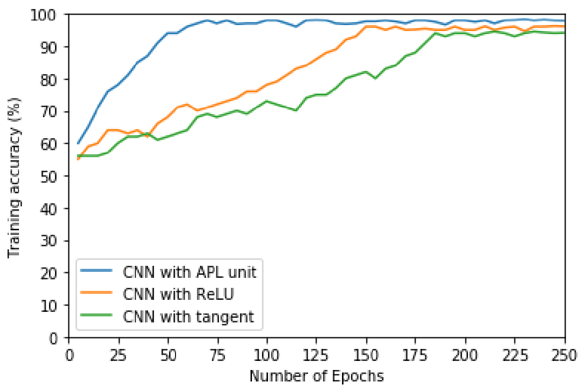

Figure 5. The training curves of the deep CNN with different activation functions.

Moreover, our network with APL units reaches the highest training accuracy around 98 % after about 65 epochs, and then it remains stable. The training accuracy of the same network with ReLUs and tangent functions is not higher than 96 %, and they become stable after about 150 and 200 epochs respectively. This means that our network consumes fewer computing resources and works faster to learn features of fire and smoke when it has APL units in hidden layers than when it has ReLUs and tangent functions. The detailed training process of our network with different activation functions is illustrated in Fig. 5.

### 4) *Experiments with data augmentation*

Experimental results in the previous section show that the Accuracy Rate of our network reaches only around 94% and there is a considerable FAR error. One of the reasons for the poor performance of the network is the overfitting problem. This means that despite the fact that training accuracy becomes very high after only 50 epochs, a network trained with only 2325 images does not perform well on test data. To solve this issue, we train the network with other datasets in which we increase the number of images using different data augmentation techniques. This idea helps our network to achieve better performance.

The testing results listed in Table VI show that when we train our network on the larger datasets we can achieve better accuracy and detection rates as well as a lower error rate. For example, when we train the network on Set2, AR increases from 94.85% to 95.87%, and DR increases from 93.12% to 93.41%, while there is minor drop in FAR from 0.63% to 0.60%. This shows that when a network is trained on larger, balanced datasets which include images of smoke and fire in different weather conditions and seasons, it can learn the features of fire and smoke better. Experimental results of our deep CNN trained on Set3, which consists of images generated using GANs, are better than those the network trained on the original dataset, Set1.

TABLE VI. EXPERIMENTS WITH DATA AUGMENTATION

| **Datasets** | AR (%) | DR (%) | FAR (%) |
|---|---|---|---|
| Set1 | 94.85 | 93.12 | 0.63 |
| Set2 | 95.87 | 93.41 | 0.60 |
| Set3 | 95.21 | 93.91 | 0.58 |
| Set4 | **97.15** | **96.18** | **0.33** |

Inspired by the progress made using data augmentation techniques, we decided to train our deep CNN using Set4. This dataset is created by combining two data augmentation techniques and it consists of 12830 images. When we evaluate results on test data, as shown in Table VI, we achieve considerably better results compared with the results of the network trained on smaller sets. Using data augmentation techniques improved the accuracy rate and detection rate by more than 2% and 3% respectively. Finally, using Set4, we achieve a 97.15% accuracy rate, a 96.18% detection rate, and a very low 0.33% false alarm rate. Experimental results indicate that the overfitting problem can be solved when we generate more images using data augmentation techniques and generate larger, balanced datasets to train the network.

### 5) *Comparisons with Other Deep CNNs*

To show the advantages of our method and the usefulness of using data augmentation in our task, we compare our deep CNN method with several other deep CNN methods that performed well on visual recognition problems. We compare the evaluation performance of our method with the prototypical deep CNN network, AlexNet [7], and the other well-known networks ResNet [14] (ILSVRC' 15) and DenseNet [17] that achieved excellent performance on the ImageNet challenge. We implement all networks by ourselves and train on both the original and the augmented data. We use different sizes of images for different network architectures, but the mini-batch size was set to 64 for all networks. All training-relevant hyper-parameters of our network are as listed in Table II.

All networks are trained using both the original images in *Set1*, and the augmented images in *Set4*; *Set5* is used for validation and *Set6* for testing. Although our network does not show the best results compared with other networks when it trains on a small number of images, its performance improves when data augmentation comes into play. For example, our deep CNN trained on the augmented dataset achieves the highest AR and the lowest FAR of all four networks, although DenseNet [17] achieves better DR than our network in this case. Without data augmentation, the only metric on which our deep CNN performs best is FAR.

TABLE VI. COMPARISONS WITH OTHER DEEP CNNS

| **Deep CNNs** | **Input image size** | **Original data** | | | **Augmented data** | | |
|---|---|---|---|---|---|---|---|
| | | AR (%) | DR (%) | FAR (%) | AR (%) | DR (%) | FAR (%) |
| AlexNet | 227 | 93.41 | 92.85 | 0.86 | 96.15 | 95.25 | 0.57 |
| ResNet | 32 | 93.91 | **93.15** | 0.72 | 96.82 | 95.81 | 0.35 |
| DenseNet | 224 | **94.66** | 92.31 | 0.66 | 97.05 | **96.57** | 0.49 |
| Our method | 128 | 94.85 | 93.12 | **0.63** | **97.15** | 96.18 | **0.33** |

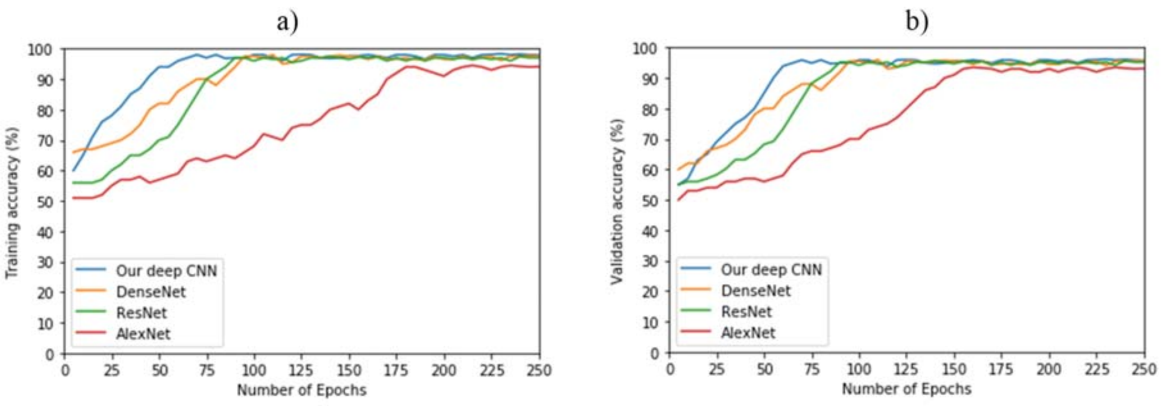

Figure 6. Training and validation curves of our deep CNN, DenseNet, ResNet, and AlexNet. a) The training curves, b) The validation curves.

Training and validation processes of all networks are shown in Fig. 6. We illustrate the training curves using the augmented dataset, *Set4*, and validation using *Set5*. In both cases, our network achieves the highest accuracy at the best speed of all the networks (around 65 epochs for our network, whereas DenseNet and ResNet require around 100 epochs to achieve 98% training accuracy). As AlexNet learns a vast number of parameters, it needs almost 180 epochs to achieve 97% training and 96 % validation accuracy.

*6) Comparisons with other fire and smoke recognition methods*

As we mentioned in Section I, there have been conducted little research on fire and smoke image recognition using deep CNNs. Moreover, all of the existing CNN algorithms are capable of classifying either only smoke [9] or fire [6] from natural images. In this section we explain the difference and the similarities between Deep Normalization and Convolutional Neural Networks (DNCNN) for smoke detection [9] and our method for fire and smoke image classification.

The main contribution in the DNCNN algorithm is that traditional convolutional layers have been replaced with normalization and convolutional layers, whereas we used normalization technique for only input images. Another considerable difference between our approach and DNCNN is that we found APL units efficient as activation functions for hidden layers of the network, while authors of the opposite work claim that ReLUs demonstrate better performance for smoke recognition task.

Despite the fact that DNCNN is developed for recognizing only smoke images, we modified the output layer of the network to be able to perform 3-way - fire, smoke and background image classification. Both our work and this algorithm have been trained in the same environment, using the largest dataset, *Set4* (Table 3) and validated using *Set5*. Experimental results and other hyper-parameters of the networks are illustrated in Table VIII.

TABLE VIII. COMPARISONS BETWEEN OUR APPROACH AND DNCNN

| **Algorithms** | Input image size | Training time per epoch (s) | AR (%) | DR (%) | FAR (%) |
|---|---|---|---|---|---|
| DNCNN | 48 | **54** | 92.12 | 90.31 | 1.15 |
| Our method with ReLU | 128 | 73 | 94.25 | 92.06 | 0.92 |
| Our method with APL | 128 | 69 | **97.15** | **96.18** | **0.33** |

The experimental results in Table VIII demonstrate that DNCNN consumes less time for per training epoch than both versions of our method. This is because of the input image size and as well as the impact of normalization technique in DNCNN algorithm. On the other hand, our method with APL units demonstrate the best evaluation performance in the test data by achieving the best accuracy and detection rates as well as the lowest false alarm rate, which are the main factors of the recognition algorithms.

Overall, comparisons of our deep CNN network with other approaches show that our method achieves better performance in fire and smoke image classification tasks when trained and tested with the data we provided.

*7) Discussion*

Fire and smoke recognition algorithms are generally used to warn possible disasters in order to avoid huge damages. Therefore, these types of methods should be very efficient and robust in detecting fire or flames, as well as should prevent false alarms. One major problem we face during the development and generalization of our algorithm is that images with huge cloud of dust can be mistakenly classified as smoke because of the visual similarities between dust and smoke. To solve this misclassification problem and reduce the false alarm rates we collected 200 dust images and augmented using the techniques explained in Section II. After augmentation procedure, we have new 1150 dust images and 1000 of these images have been added as background images to *Set4* and new dataset named *Set4.1* has been created. Remaining images are added to the *Set6*, in order to create new test data, namely *Set6.1*.

We retrained the network using newly generated *Set4.1* and tested using *Set6.1*. Experimental results in the Table

IX show that adding dust images to the training data decreased the false alarm rates.

TABLE IX. COMPARISONS OF FAR RESULTS OF THE NETWORK TRAINED WITH DIFFERENT DATASETS

| **Training data** | Total number of training images | Number of dust images in training dataset | FAR (%) |
|---|---|---|---|
| *Set4* | 12830 | 0 | 11.53 |
| *Set4.1* | 13830 | 1000 | 3.56 |

Evaluation results in the Table IX demonstrate that, the network trained without dust images as background class performs very bad by misclassifying more than 11 % of dust images. Adding 1000 dust images in training helped to solve this problem, by decreasing FAR to 3.56 %.

## V. CONCLUSION

In this paper, we showed that very high classification performance can be achieved using deep CNNs, even when limited data is provided. The overfitting problem, caused by a limited set of image data for training, leads to poor performance in neural network models. To solve this problem, we enlarge our training sets using various data augmentation techniques. The use of GANs to create additional training samples improves the diversity of data available, and helps the network to learn fire and smoke features under different weather and light (day-time and night-time) conditions. Furthermore, applying another activation function, the adaptive piecewise linear function, facilitates notable performance improvements in our deep CNN model, without significantly increasing the number of learnable parameters.

In the future, we intend to expand on our current work and develop a robust fire and smoke detection algorithm capable of pinpointing the location of fire and smoke.

## ACKNOWLEDGEMENTS

This research was supported by the Gachon University project (project number is 2015-0046), by the Ministry of Trade Industry and Energy “Development of Knowledge/Intelligent DB Sharing Technologies” 2016-659 from 2017 (project number is 10077965), and by the MSICT(Ministry of Science and ICT), Korea, under the ITRC(Information Technology Research Center) support program(IITP-2018-2017-0-01630) supervised by the IITP(Institute for Information & communications Technology Promotion).

## REFERENCES

[1] Chen, Thou-Ho, Yen-Hui Yin, Shi-Feng Huang, and Yan-Ting Ye. "The smoke detection for early fire-alarming system base on video processing." Intelligent Information Hiding and Multimedia Signal Processing, pp. 427-430, 2006. doi:10.1109/IIH-MSP.2006.265033

[2] Töreyin, B. Uğur, Yiğithan Dedeoğlu, Uğur Güdükbay, and A. Enis Cetin. "Computer vision based method for real-time fire and flame detection." Pattern recognition letters 27, no. 1, pp. 49-58, 2006. doi: 10.1016/j.patrec.2005.06.015

[3] Mueller, Martin, Peter Karasev, Ivan Kolesov, and Allen Tannenbaum. "Optical flow estimation for flame detection in videos." IEEE Transactions on image processing 22, no. 7, pp.2786-2797, 2013. doi:10.1109/TCSVT.2015.2392531

[4] Bugaric, M., Jakovcevic, T., & Stipanicev, D. Computer Vision Based Measurement of Wildfire Smoke Dynamics. Advances in Electrical and Computer Engineering, 2015. Volume 15, no 1, 55-62. doi 10.4316/AECE.2015.01008

[5] Çelik, Turgay, Hüseyin Özkaramanlı, and Hasan Demirel. "Fire and smoke detection without sensors: Image processing based approach." Signal Processing Conference, 2007 15th European, pp. 1794-1798, 2007.

[6] Zhang, Qingjie, Jiaolong Xu, Liang Xu, and Haifeng Guo. "Deep convolutional neural networks for forest fire detection." Proceedings of the 2016 International Forum on Management, Education and Information Technology Application. Atlantis Press. 2016.

[7] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems, pp. 1097-1105, 2012. doi: 10.1145/3065386

[8] Tao, Chongyuan, Jian Zhang, and Pan Wang. "Smoke detection based on deep convolutional neural networks." In Industrial Informatics-Computing Technology, Intelligent Technology, Industrial Information Integration (ICIICII), 2016 International Conference on, pp. 150-153, 2016. doi: 10.1109/ICIICII.2016.0045

[9] Yin, Zhijian, Boyang Wan, Feiniu Yuan, Xue Xia, and Jinting Shi. "A deep normalization and convolutional neural network for image smoke detection." IEEE Access 5, pp. 18429-18438, 2017. doi: 10.1109/ACCESS.2017.2747399

[10] Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial nets." Advances in neural information processing systems, pp. 2672-2680. 2014.

[11] LeCun, Yann, Léon Bottou, Yoshua Bengio, and Patrick Haffner. "Gradient-based learning applied to document recognition." Proceedings of the IEEE 86, no. 11, pp, 2278-2324, 1998. doi: 10.1109/5.726791

[12] Zeiler, Matthew D., and Rob Fergus. "Visualizing and understanding convolutional networks." European conference on computer vision, pp. 818-833, 2014. doi: 10.1007/978-3-319-10590-1_53

[13] He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification." Proceedings of the IEEE international conference on computer vision, pp. 1026-1034. 2015. doi: 10.1109/ICCV.2015.123.

[14] He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770-778. 2016. doi: 10.1109/CVPR.2016.90.

[15] Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. "Going deeper with convolutions." Cvpr, 2015. doi: 10.1109/CVPR.2015.7298594.

[16] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv: 1409.1556, 2014

[17] Huang, Gao, Zhuang Liu, Kilian Q. Weinberger, and Laurens van der Maaten. "Densely connected convolutional networks." Proceedings of the IEEE conference on computer vision and pattern recognition, vol. 1, no. 2, p. 3. July, 2017. doi: 10.1109/CVPR.2017.243

[18] Simard, Patrice Y., David Steinkraus, and John C. Platt. "Best practices for convolutional neural networks applied to visual document analysis." ICDAR, vol. 3, pp. 958-962, 2003. doi: 10.1109/ICDAR.2003.1227801.

[19] Zhu, Jun-Yan, Taesung Park, Phillip Isola, and Alexei A. Efros. "Unpaired image-to-image translation using cycle-consistent adversarial networks." arXiv preprint arXiv:1703.10593, 2017. doi: 10.1109/ICCV.2017.244.

[20] Isola, Phillip, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. "Image-to-image translation with conditional adversarial networks." arXiv preprint, 2017. doi: 10.1109/CVPR.2017.632.

[21] Agostinelli, Forest, Matthew Hoffman, Peter Sadowski, and Pierre Baldi. "Learning activation functions to improve deep neural networks." arXiv preprint arXiv:1412.6830, 2014. doi: 10.1140/epjc/s10052-011-1554-0.

[22] Glorot, Xavier, and Yoshua Bengio. "Understanding the difficulty of training deep feedforward neural networks." Proceedings of the thirteenth international conference on artificial intelligence and statistics, pp. 249-256, 2010.

[23] Yuan, Feiniu, Jinting Shi, Xue Xia, Yuming Fang, Zhijun Fang, and Tao Mei. "High-order local ternary patterns with locality preserving projection for smoke detection and image classification." Information Sciences 372, p.p: 225-240. doi: 10.1016/j.ins.2016.08.040.

[24] Abadi, Martín, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado et al. "Tensorflow: Large-scale machine learning on heterogeneous distributed systems." arXiv preprint arXiv:1603.04467, 2016