Experimental Exploration of Compact Convolutional Neural Network Architectures for Non-temporal Real-time Fire Detection

Ganesh Samarth C.A.[1,2], Neelanjan Bhowmik[2], Toby P. Breckon[2,3]
[1]Department of Electrical Engineering, Indian Institute of Technology Dharwad, India
Department of {Computer Science[2] | Engineering[3]}, Durham University, UK

***Abstract*—In this work we explore different Convolutional Neural Network (CNN) architectures and their variants for non-temporal binary fire detection and localization in video or still imagery. We consider the performance of experimentally defined, reduced complexity deep CNN architectures for this task and evaluate the effects of different optimization and normalization techniques applied to different CNN architectures (spanning the Inception, ResNet and EfficientNet architectural concepts). Contrary to contemporary trends in the field, our work illustrates a maximum overall accuracy of 0.96 for full frame binary fire detection and 0.94 for superpixel localization using an experimentally defined reduced CNN architecture based on the concept of InceptionV4. We notably achieve a lower false positive rate of 0.06 compared to prior work in the field presenting an efficient, robust and real-time solution for fire region detection.**

***Index Terms*—binary fire detection, real-time, non-temporal, reduced complexity, deep convolutional neural network, super-pixel, localization**

I. Introduction

Automated fire detection and localization have become essential tasks in the modern day auto-monitoring systems. The increasing prevalence of industrial, public space and general environment monitoring using security-driven CCTV video systems has given rise to the consideration of these systems as sources of initial fire detection. Furthermore, the on-going consideration of remote vehicles for fire detection and monitoring tasks [1]–[3] further enhances the demand for autonomous fire detection from such platforms. Fire detection stands out among other object classification tasks as fire does not have a definite shape or pattern but instead varies with the underlying material composition.

Most early work revolves around a color, texture and shape based approach to fire detection and localization. A color threshold based approach is explored in [4] which is extended with the basic consideration of motion by [5]. Later work considers the temporal variation of fire in the Fourier domain [6] with progressive studies formulating the problem as a Hidden Markov Model [7]. More recent works consider machine learning based classification approaches to the fire detection problem [3], [8], [9]. Chenebert et al. [9] consider the use of a non-temporal approach along with colour and texture feature descriptions as an input to a shallow neural network classifier.

With the advent of deep learning, further developments to fire detection based on deep CNN architectures now perform binary fire detection more efficiently and robustly compared to earlier color based approaches [4], [10]. Attempts to detect fire based on robust smoke detection [11] have been made by using synthetically produced smoke images. Some recent work [12] applies YOLO architecture [13] to perform flame detection. There has also been attempt to develop custom architectures [14] involving convolution, fully connected and pooling layers on a custom dataset created using Generative Adverserial Networks (GAN). The work of [15] considers deep CNN architectures such as VGG16 [16] and ResNet50 [17] for the fire detection task. A further experimental approach to fire detection [18] is based on exploring InceptionV1 [19] and AlexNet [20] architectures.

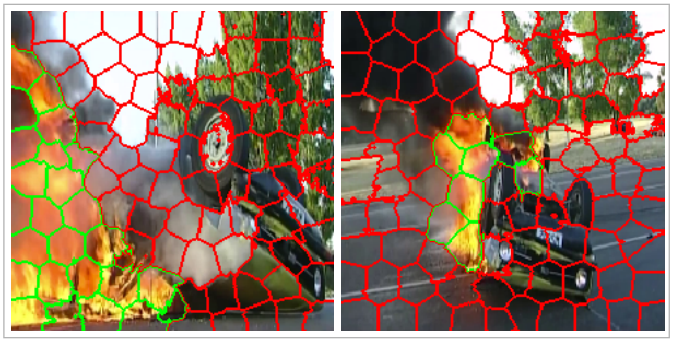

Fig. 1. Example fire detection and localization using superpixel (fire = green, no-fire = red).

The work of [22], a direct precursor to this study, explored fire detection and localization based on both full-frame binary fire detection and superpixels (Figure 1) based on similar experimentally defined CNN architectural variants derived from the seminal InceptionV1 [19] and AlexNet [20] architectures (InceptionV1-OnFire, FireNet, [22]). InceptionV1-OnFire achieved 0.89 detection accuracy for superpixel based detection whilst FireNet achieves 0.92 for full frame binary fire detection.

In this work we expand upon the study of [22] considering a similar experimental approach to the definition of reduced complexity CNN but based on contemporary advances in the Inception [21], [23] and ResNet [17], [23] architectures. This is a non-temporal approach to the fire detection problem which is highly suited for non-stationary fire detection scenarios.

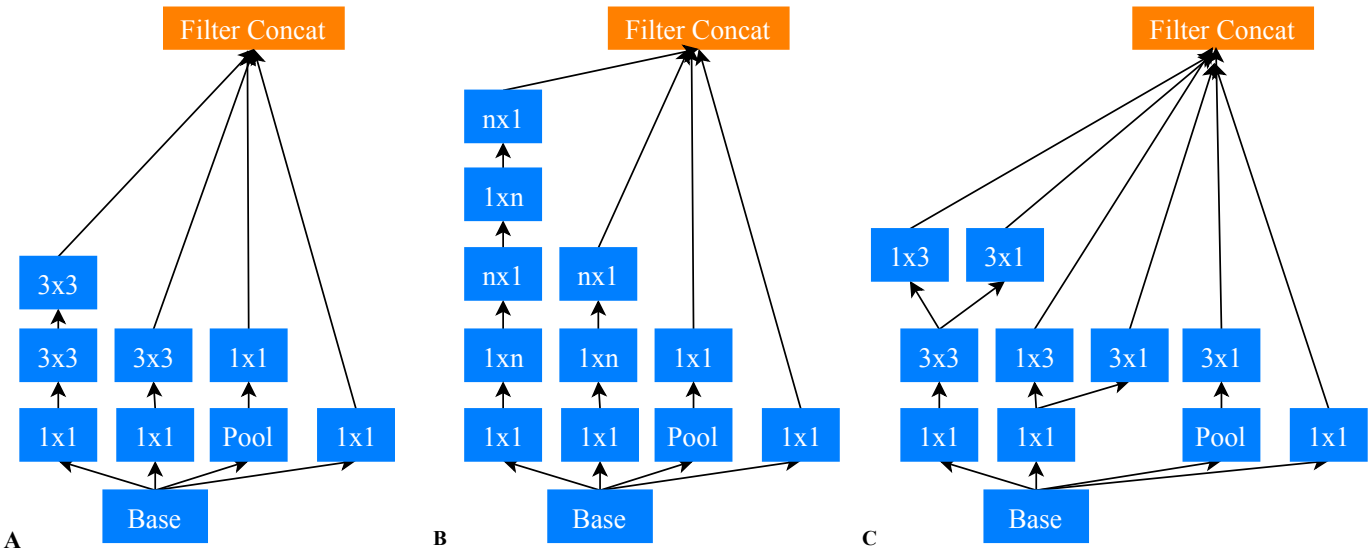

Fig. 2. Three variants of InceptionV2 [21] : Module-A with 3×3 kernels (A), Module-B with asymmetric convolutions with $n = 7$ (B), and Module-C with the wide filter banks of 1×3 and 3×1 kernels (C).

We explicitly consider two fire detection problems: (a) binary fire detection to determine if fire is present in a particular frame and (b) in-frame superpixel localization to determine the precise location of fire within that frame.

II. Proposed Approach

Our approach experimentally defines CNN architectures with low complexity to address the fire detection tasks identified. Whilst prior work [22] focuses only on AlexNet and InceptionV1 variants, here we expand this remit to additional reference CNN architectures.

A. Reference Architectures

InceptionV2 [21] is inspired from InceptionV1 [19]. The intuition is that, reducing the dimensions drastically may cause loss of information, known as a "representational bottleneck". Smart factorization techniques are used to make convolutions more efficient in terms of computational complexity. Hence three different variants of the inception modules are defined (Module - A, B, C). In Module-A, the filters with a 5×5 kernel size are replaced with two 3×3 kernels (Figure 2A). A 5×5 convolution is 2.78 times more expensive than a 3×3 convolution. Hence, two 3×3 convolutions are connected which leads to a boost in performance. Moreover $n \times n$ convolutions can be further factorized to a combination of $1 \times n$ and $n \times 1$ convolutions. This is found to be three times more efficient and is implemented as the Module-B of the inception variant with $n = 7$ (Figure 2B). Filter banks are further expanded to remove the representational bottleneck. The third variant of the inception module, i.e. Module-C (Figure 2C), has 3×3 filters factorized into parallel 1×3 and 3×1 filters and merged to make the module wider. The stem of the network consists of two convolution layers followed by a pooling layer and by three convolution layers.

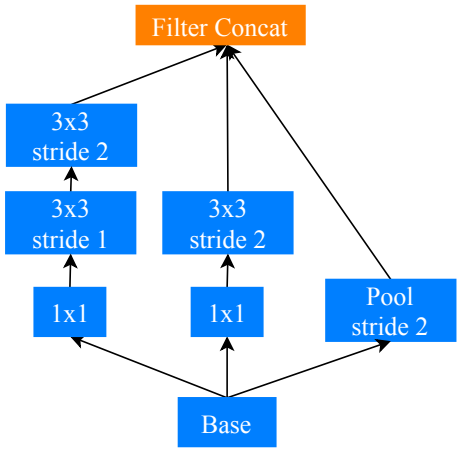

Fig. 3. Grid size reduction by connecting convolution layer and max-pooling in parallel in InceptionV3 [21].

The inception modules are connected with three modules of Module-A, five of Module-B and two of Module-C. This is followed by global max pooling, a linear and a final softmax layer.

InceptionV3 [21] is a very similar architecture to InceptionV2 and it uses same three modular components (Figure 2) additional features, such as Root Mean Square Propagation (RMSProp), label smoothing and batch normalization. Grid size reduction (Figure 3) is introduced in this model, where feature maps are connected with a convolution layer of stride two and a max-pooling layer in parallel for concatenation. This is shown better when compared to just using pooling to reduce the dimensions as this leads to some loss of information. The first and second variants of the inception modules have a reduction block succeeding them.

InceptionV4 [23] is the next version of Inception proposed

(with three variants Module - A,B,C), which differs from its previous version only with respect to its stem. InceptionV4 utilizes the idea of efficient grid size reduction (Figure 3) in its stem to reduce the dimensions of the image without any significant loss of information before being passed to the inception and its reduction blocks.

ResNet [17] and variants show very promising performance for the object recognition task. Deep networks are hard to train due to the notorious vanishing gradient problem [20]. Since the gradient is back-propagated to earlier layers, repeated multiplication may make the gradient infinitively small. As a result, when network becomes deeper, its performance may degrade rapidly. ResNet is based on the idea of skip connections which ensures optimal hyperparameter values such as number of layers and overcomes the vanishing gradient problem.

Inception-ResNet [23] as the name suggests is a hybrid architecture of Inception and ResNet. Essentially, the pooling operations in the inception modules have been replaced with residual connections. However these changes are only to the Inception blocks and the reduction blocks remain intact. There are two variants of Inception-ResNet network, denoted as v1 and v2, whose only difference lies in the hyperparameter settings. Inception-ResNet has been shown to achieve superior performance with a lower number of training epochs.

EfficientNet [24] is one of the most recent architectures based on a novel scaling method that uses compound coefficient to scale networks. Unlike conventional approaches that arbitrarily scale network dimensions, such as width, depth and resolution, this method uniformly scales each dimension with a fixed set of scaling coefficients. Based on this scaling method and advancements in NAS lead to the development of a family of models known as EfficientNet which offers a ten fold efficiency gain when compared to the state of the art for ImageNet classification [25].

B. Simplified CNN Architectures

Our experimental approach systematically investigated variations in architectural configurations of each reference architecture. Performance is measured using the same evaluation parameters set out in Section III.

For InceptionV2 we consider three different variants and four different sub-variants. Each of the three variants consists only of the inception modules illustrated in Figure 2 (Modules A-C) to facilitate separate evaluation. Since the primary goal of this work is to develop a simplified CNN architecture, we restrict the maximum number of inception modules to six in each of the three major variants. The various network variants evaluated for each of the major variants are denoted as follows:

- *A3-A6* - A variant consisting of only Module-A components, where A contains n modules for $n = \{3..6\}$.
- *B3-B6* - consisting of only Module-B components which have asymmetric factorization of 7×7 convolutions, where B contains n modules for $n = \{3..6\}$.
- *C3-C6* - consisting of only Module-C components which are broadened and concatenated with n modules for $n = \{3..6\}$.

InceptionV3 is architecturally modified into 12 different variants with the naming convention $InceptionV3_{v01-12}$. Each of these variants use different combinations of inception modules (Figures 2/3). The first six variants have the same number of filters as mentioned in the original work [21] and the latter six variants have reduced number of filters according to Eq. 1. We restrict the number filters in each layer less than 100, and secondly, the number of filters is a multiple of the original number of filters as in the original work [21]. If the original number of filter is M in a layer, then the reduced number of filter (M_r) is calculated as follow:

$$M_r = \begin{cases} \left\lfloor \frac{M}{2^{\lceil \log_2 \frac{M}{100} \rceil}} \right\rceil & M > 100 \\ M & otherwise \end{cases} \quad (1)$$

Each variation of InceptionV3, which is a combination of Module A/B/C, grid size reduction (GR) of Module - A/B and reduced number of filters applied (according to Eq. 1) and connected one upon the other, is presented in Table I. Similar to InceptionV3, InceptionV4 is modified into 12 different variants. As with InceptionV3, the first six variants ($v01-06$) consist of the network variants with the same number of filters as mentioned in the original work for InceptionV4 [23] and the latter six variants ($v07-12$) consist of both reduced number of filters with a modified stem to reduce the computational complexity. The InceptionV4 variants also follow the similar naming convention as InceptionV3 with variants being named as $InceptionV4_{v01-12}$. Each variants of InceptionV4 is presented in Table II.

ResNet has been evaluated as it is with varying depths. The four ResNet models are ResNet-{*18,34,50,101*}. Inception-ResNet v1 and v2 have been evaluated with no modifications. Three different variations of EfficientNet-{*B0,B1,B2*}, defined as in the original work [24], has been evaluated.

Based on an exhaustive set of experimentation over the full set of variants outlined, under the conditions outlined in Section II, we experimentally identify and propose the following two maximally reduced complexity performing architectural variants targeted towards the fire detection and localization task.

InceptionV3-OnFire is inspired by the performance of the $InceptionV3_{v09}$ variant. One of each Inception Module - A, B, and C are connected to develop the InceptionV3-OnFire architecture, as illustrated in Figure 4. To reduce the complexity, the number of filters in each layer are restricted as according to the Eq. 1.

InceptionV4-OnFire is a three layered version of InceptionV4 which is based on the $InceptionV4_{v05}$ variant containing one each of the three inception Module- A, B and C (Figure 5). The grid size reduction module is removed. Each of the inception modules followed the same definition as that of the original work. A dropout of 0.4 is applied at the end

TABLE I
INCEPTIONV3 VARIANTS WITH DIFFERENT COMPONENTS.

Architecture	Module-A	GR-A	Module-B	GR-B	Module-C	Reduced filter
$InceptionV3_{v01}$	✓	✓	✓	✓	✓	
$InceptionV3_{v02}$	✓		✓	✓	✓	
$InceptionV3_{v03}$	✓		✓		✓	
$InceptionV3_{v04}$		✓	✓	✓	✓	
$InceptionV3_{v05}$		✓	✓	✓		
$InceptionV3_{v06}$		✓	✓	✓		
$InceptionV3_{v07}$		✓	✓	✓		✓
$InceptionV3_{v08}$	✓	✓	✓	✓		✓
$InceptionV3_{v09}$	✓		✓		✓	✓
$InceptionV3_{v10}$		✓	✓	✓	✓	✓
$InceptionV3_{v11}$	✓		✓	✓	✓	✓
$InceptionV3_{v12}$	✓	✓	✓	✓		✓

TABLE II
INCEPTIONV4 VARIANTS WITH DIFFERENT COMPONENTS.

Architecture	Module-A	GR-A	Module-B	GR-B	Module-C	Reduced filter
$InceptionV4_{v01}$	✓	✓	✓	✓	✓	
$InceptionV4_{v02}$	✓	✓	✓		✓	
$InceptionV4_{v03}$	✓		✓	✓	✓	
$InceptionV4_{v04}$	✓	✓		✓	✓	
$InceptionV4_{v05}$	✓		✓		✓	
$InceptionV4_{v06}$		✓	✓	✓	✓	
$InceptionV4_{v07}$	✓	✓	✓	✓	✓	✓
$InceptionV4_{v08}$	✓	✓	✓	✓		✓
$InceptionV4_{v09}$	✓		✓	✓	✓	✓
$InceptionV4_{v10}$		✓	✓	✓	✓	✓
$InceptionV4_{v11}$	✓		✓	✓		✓
$InceptionV4_{v12}$	✓	✓	✓	✓	✓	✓

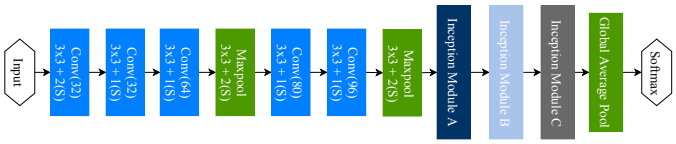

Fig. 4. Reduced complexity architecture for InceptionV3-OnFire optimized for fire detection.

of the network to prevent the model from over-fitting. The same stem as in the original InceptionV4 architecture is used as illustrated in Figure 5.

Overall, the governing intuition based on these variants is that, a combination of all the three inception modules (Figure 2) will perform better as the architecture is equipped with both depth and width to optimally learn how to detect and localize fire. The grid reduction modules are mainly used to shrink the height and width of the image in a more optimal fashion, although it eventually leads to information loss.

C. Superpixel Localization

Further expanding this work, we adopt the use of image over-segmentation based fire localization, contrary to the earlier works [5] [26] [27], which rely on colour based fire localization. Superpixel based approaches over-segment the image into perceptually meaningful regions which are similar in colour and texture. Specifically we incorporate the Simple Linear Iterative Clustering (SLIC) [28] over-segmentation approach, which performs iterative clustering in a similar manner to *k-means* to reduced spatial dimensions, where the image is segmented into approximately equally sized superpixels (Figure 1). Each over-segmented/superpixel image region is subsequently classified using proposed InceptionV3-OnFire/InceptionV4-OnFire architecture formulated as a {*fire, no-fire*}, for fire detection task. To boost the performance, we additionally use the network weight initialization via transfer learning from the primary full frame binary fire detection task.

III. EXPERIMENTAL SETUP

We address the problem of full frame binary fire detection (*i.e. is there fire present in the image as a whole - yes/no?*) as well as fire localization (*i.e location of the fire in the image?*). All networks are trained using Nvidia GeForce GTX 1080Ti GPU via TensorFlow (1.13.1 + TFLearn 0.3.2). The network variants are tested with different optimizers such as

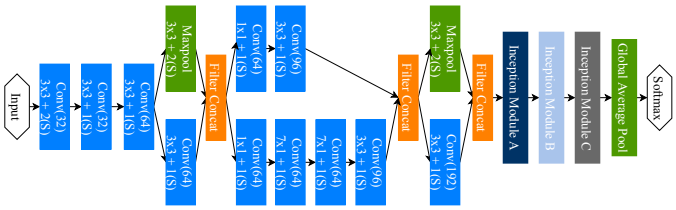

Fig. 5. Reduced complexity architecture for InceptionV4-OnFire optimized for fire detection.

stochastic gradient descent (SGD) with momentum and Root Mean Square Propagation (RMSProp) along with normalization techniques such as local response normalization and batch normalization. The training is performed with categorical cross entropy loss, for 30 epochs and a learning rate of 0.001.

A. Full-frame Binary Fire Detection

For the binary fire detection problem, network training and testing are performed on the dataset created in the work [22] which consists of 23,408 images. This dataset is split (80:20 split) into two portions for training and validation. An additional set of 2,931 images was used for cross validation.

B. Superpixel Localization Setup

To evaluate within the context of in-frame localization, we use the dataset created in the work [22]. The network architectures are trained on a total of 8,635 fire superpixel images, and 10,000 non-fire superpixel images with a test set of 3,000 images containing 1,500 fire and 1,500 non-fire examples. These images are further pre-processed to centre the superpixel region to make it location independent and padded to a size of 224×224.

IV. Evaluation

For statistically comparing different architectures we consider the true positive rate (TPR), false positive rate (FPR) along with F-score (F), Precision (P) and Accuracy (A), Complexity (number of parameters in millions, C), the ratio between accuracy and number of parameters (A:C) and achievable frames per second (fps) throughput.

The results of full-frame binary fire detection are presented in Table III. We present only the best performing variants of the reference architectures (Sec. II-A) with results shown in the Table III (middle). From the results, we can observe that our proposed variant of InceptionV4, InceptionV4-OnFire, offers the best performance (Table III, lower) in terms of accuracy, TPR (A: 0.96, TPR: 0.95) compared to other architectures. Both of our proposed architectures, InceptionV3-OnFire and InceptionV4-OnFire, achieve the lowest false positive rate (FPR: 0.07/0.04, Table III, lower), compared to previous work of FireNet [22] with (FRP: 0.09 FPR, Table III, upper). The reduced complexity, InceptionV3 variant performs just marginally worse when compared to the InceptionV4 variant but still outperforms InceptionV1-OnFire [22] in terms of accuracy, false positive rate and the accuracy is to number of parameters ratio (Table III).

TABLE III
Statistical performance for full-frame binary fire detection. Upper: Prior work. Middle: Reference architectures. Lower: Our approaches.

Architecture	TPR	FPR	F1	P	A
FireNet [22]	0.92	0.09	0.93	0.93	0.92
InceptionV1-OnFire [22]	0.96	0.10	0.94	0.93	0.93
InceptionV2-B6	0.97	0.09	0.95	0.94	0.95
ResNet-18	0.92	0.05	0.94	0.96	0.93
Inception-ResNet-v1	0.84	**0.03**	0.90	0.97	0.89
EfficientNet-B0	0.94	0.16	0.91	0.89	0.90
InceptionV3-OnFire	**0.95**	0.07	0.95	0.95	0.94
InceptionV4-OnFire	**0.95**	0.04	**0.96**	**0.97**	**0.96**

TABLE IV
Computational Efficiency for full-frame binary fire detection.

Architecture	C	A(%)	A:C	fps
FireNet [22]	68.3	91.5	1.3	**17**
InceptionV1-OnFire [22]	1.2	93.4	77.9	8.4
InceptionV3-OnFire	0.96	94.4	**98.09**	13.8
InceptionV4-OnFire	7.18	95.6	13.37	12

Conversely, we find that the InceptionV3 variant marginally outperforms the InceptionV4 variant in terms computational efficiency (A:C, fps in Table IV-lower). Although the number of parameters is reduced to 0.96 million in InceptionV3-OnFire compared to 68.3/1.2 million in FireNet/InceptionV1OnFire [22], the run-time throughput is still higher for FireNet (Table IV). Whilst FireNet [22] provides a maximal throughput of 17 fps, it is notable that InceptionV3-OnFire provides the maximal accuracy to complexity ratio.

TABLE V
Localization Results - within frame superpixel approach.

Architecture	TPR	FPR	F	P	A
InceptionV1-OnFire [22]	0.92	0.17	0.9	0.88	0.89
InceptionV3-OnFire	0.94	0.07	0.94	0.93	**0.94**
InceptionV4-OnFire	**0.94**	**0.06**	**0.94**	**0.94**	**0.94**

The results for superpixel based fire localization are presented in Table V where we can see that InceptionV4-OnFire marginally outperforms InceptionV3-OnFire in terms of a lower FPR with equal overall accuracy representing a 5% increase in performance over prior work in the field (InceptionV1-OnFire [22]).

Qualitative examples of this localization (InceptionV4-OnFire), including the canonoical challenge of red coloured non-fire regions, are illustrated in Figure 6. From this figure, we see the positive performance impact of transfer learning from the initial full-frame binary fire detection into this fire localization task. The region inside yellow dashed box in the Figure 6A is falsely detected is *fire*, however, with transfer

Fig. 6. Comparison of results in yellow dashed box without transfer learning (A) and with transfer learning (B), where fire = green, no-fire = red.

learning the same region is correctly detected as *no-fire* in Figure 6B. Transfer learning significantly reduces this type of FP occurrence by approximately 10% (FPR: 0.06 from previously 0.17, Table V).

From Tables IV and III we can see that InceptionV4-OnFire/InceptionV3-OnFire offer slightly lesser computational performance in terms of frame-rate than prior work (FireNet, [22]) but significantly improved detection performance.

V. Conclusion

Our proposed reduced complexity CNN architecture (InceptionV4-OnFire), which is experimentally defined from leading CNN architectures, achieve maximal 0.96 accuracy for binary full-frame fire detection task. We significantly reduce the false positive rate as low as 0.04 outperforming the prior state-of-the-art approach of FireNet [22]. We also manage reduce the number of parameters of InceptionV3-OnFire by 0.24 million compared to architectures in [22]. Furthermore, for superpixel based fire detection, we notably reduce the false positive rate to 0.06 by employing a transfer learning strategy. Overall, this work presents robust and reduced complexity architectures for full-frame/superpixel fire detection task enabled by extended, exhaustive experimental evaluation.

References

[1] A. Bardshaw, "The UK security and fire fighting advanced robot project," in *IEE Coll. on Advanced Robotic Initiatives in the UK*, London, UK, 1991.

[2] J. Martinezdedios, L. Merino, F. Caballero, A. Ollero, and D. Viegas, "Experimental results of automatic fire detection and monitoring with UAVs," *Forest Ecology and Management*, vol. 234, pp. S232–S232, Nov. 2006.

[3] D. Zhang, S. Han, J. Zhao, Z. Zhang, C. Qu, Y. Ke, and X. Chen, "Image based forest fire detection using dynamic characteristics with artificial neural networks," in *International Joint Conference on Artificial Intelligence*, 2009, pp. 290–293.

[4] G. Healey, D. Slater, T. Lin, B. Drda, and A. Goedeke, "A system for real-time fire detection," in *Proc. Int. Conf. Comp. Vis. and Pat. Rec.*, 1993, pp. 605–606.

[5] W. Phillips, M. Shah, and N. da Vitoria Lobo, "Flame recognition in video," *Pat. Rec. Letters*, vol. 23, no. 1-3, pp. 319–327, 2002.

[6] C. Liu and N. Ahuja, "Vision based fire detection," in *Proc. Int. Conf. on Pattern Recognition*, 2004, pp. 134–137.

[7] B. Toreyin, Y. Dedeoglu, and A. Cetin, "Flame detection in video using hidden Markov models," in *Proc. Int. Conf. on Image Proc.*, 2005, pp. II–1230.

[8] B. Ko, K. Cheong, and J. Nam, "Fire detection based on vision sensor and support vector machines," *Fire Safety J.*, vol. 44, no. 3, pp. 322–329, Apr. 2009.

[9] A. Chenebert, T. P. Breckon, and A. Gaszczak, "A non-temporal texture driven approach to real-time fire detection," in *Proc. Int. Conf. on Image Proc.*, Sep. 2011, pp. 1741–1744.

[10] T. Chen, P. Wu, and Y. Chiou, "An early fire-detection method based on image processing," in *Proc. Int. Conf. on Image Proc.*, 2004, pp. 1707–1710.

[11] G. Xu, Y. Zhang, Q. Zhang, G. Lin, and J. Wang, "Deep domain adaptation based video smoke detection using synthetic smoke images," *Fire safety journal*, vol. 93, pp. 53–59, 2017.

[12] D. Shen, X. Chen, M. Nguyen, and W. Q. Yan, "Flame detection using deep learning," in *International Conference on Control, Automation and Robotics*, 2018, pp. 416–420.

[13] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. Computer Vision and Pattern Recognition*, 2016, pp. 779–788.

[14] A. Namozov and Y. Im Cho, "An efficient deep learning algorithm for fire and smoke detection with limited data," *Advances in Electrical and Computer Engineering*, vol. 18, no. 4, pp. 121–129, 2018.

[15] J. Sharma, O.-C. Granmo, M. Goodwin, and J. T. Fidje, "Deep convolutional neural networks for fire detection in images," in *International Conference on Engineering Applications of Neural Networks*. Springer, 2017, pp. 183–193.

[16] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations*, 2015.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[18] K. Muhammad, J. Ahmad, I. Mehmood, S. Rho, and S. W. Baik, "Convolutional neural networks based fire detection in surveillance videos," *IEEE Access*, vol. 6, pp. 18 174–18 183, 2018.

[19] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[20] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.

[21] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. Computer Vision and Pattern Recognition*, 2016, pp. 2818–2826.

[22] A. J. Dunnings and T. P. Breckon, "Experimentally defined convolutional neural network architecture variants for non-temporal real-time fire detection," in *Proc. Int. Conf. on Image Proc.*, 2018, pp. 1558–1562.

[23] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *AAAI Conference on Artificial Intelligence*, 2017.

[24] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International Conference on Machine Learning*, 2019, pp. 6105–6114.

[25] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, and A. Berg, "Imagenet large scale visual recognition challenge," *Int. J. of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[26] T. Celik and H. Demirel, "Fire detection in video sequences using a generic color model," *Fire Safety J.*, vol. 44, no. 2, pp. 147–158, Feb. 2009.

[27] F. Yuan, "A double mapping framework for extraction of shape-invariant features based on multi-scale partitions with adaboost for video smoke detection," *Pattern Recognition*, vol. 45, no. 12, pp. 4326–4336, 2012.

[28] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "Slic superpixels compared to state-of-the-art superpixel methods," *IEEE Trans Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2274–2282, 2012.