FireWarn: Fire Hazards Detection Using Deep Learning Models

Isaac Hogan*
Donghao Qiao*
Ruikang Luo*
Mojtaba Moattari*
Austin Carthy*
Farhana Zulkernine
Queen’s University

François Rivest
Royal Military College of Canada

Mélanie Breton
DRDC – Valcartier Research Centre

Institute of Electrical and Electronics Engineers (IEEE)
International Conference on Cognitive Machine Intelligence 2021
December 13, 2021
University of Pittsburgh, Pittsburgh, PA, USA

Date of Publication from External Publisher: December 2021

Terms of Release: This document is approved for public release.

The body of this CAN UNCLASSIFIED document does not contain the required security banners according to DND security standards. However, it must be treated as CAN UNCLASSIFIED and protected appropriately based on the terms and conditions specified on the covering page.

* These authors contributed equally to the work.
IMPORTANT INFORMATIVE STATEMENTS

This document was reviewed for Controlled Goods by Defence Research and Development Canada using the Schedule to the Defence Production Act.

Disclaimer: This document is not published by the Editorial Office of Defence Research and Development Canada, an agency of the Department of National Defence of Canada but is to be catalogued in the Canadian Defence Information System (CANDIS), the national repository for Defence S&T documents. Her Majesty the Queen in Right of Canada (Department of National Defence) makes no representations or warranties, expressed or implied, of any kind whatsoever, and assumes no liability for the accuracy, reliability, completeness, currency or usefulness of any information, product, process or material included in this document. Nothing in this document should be interpreted as an endorsement for the specific use of any tool, technique or process examined in it. Any reliance on, or use of, any information, product, process or material included in this document is at the sole risk of the person so using it or relying on it. Canada does not assume any liability in respect of any damages or losses arising out of or in connection with the use of, or reliance on, any information, product, process or material included in this document.

© Her Majesty the Queen in Right of Canada (Department of National Defence), and Queen's University, 2021
© Sa Majesté la Reine en droit du Canada (Ministère de la Défense nationale), et Queen’s University, 2021

CAN UNCLASSIFIED
FireWarn: Fire Hazards Detection Using Deep Learning Models

Isaac Hogan*, Donghao Qiao*, Ruikang Luo*, Mojtaba Moattari*, Austin Carthy*, Farhana Zulkernine*, François Rivest†*, Mélanie Breton†

*School of Computing, Queen’s University, Kingston, Canada
†Department of Mathematics and Computer Science, Royal Military College of Canada, Kingston, Canada
‡Defence Research and Development Canada, Valcartier, Canada

{isaac.hogan, d.qiao, ruikang.luo, mojtaba.moattari, austin.carthy, farhana.zulkernine}@queensu.ca,
francois.rivest@{rmc.ca, mail.mcgill.ca}, melanie.breton@ecn.forces.gc.ca

Abstract—Hazardous situations such as house, car, or forest fires may be recorded by cameras long before they are identified by people. To test whether deep learning could be used to quickly detect fires, we performed a series of experiments to detect the presence of fire or smoke in images and labeled them with bounding boxes. Two custom datasets were created in this research: a fire image classification dataset, and a fire and smoke detection and localization dataset. The first one only classifies the whole image, while the detection set further provides information about where the fire or smoke is within the image. We explore the efficacy of a basic convolutional classification neural network, which proved effective for fire classification, but show that pretrained classification models such as ResNet improves the accuracy when classifying fire and non-fire images. The pretrained model achieves 97.14% testing accuracy on our fire classification dataset. For fire detection and localization, three models were trained on images of fire and smoke to find and label the regions of interest. Results show that Faster R-CNN did not perform very well on fire detection and localization, while EfficientDet and YoloV5 performed much better. Moreover, YoloV5 using low resolution images also performed well on smoke detection and localization, which is more difficult than fire. YoloV5 achieved an average precision of 46.6 on our fire and smoke detection dataset.

Keywords—Machine learning, computer vision, object detection, fire detection

I. INTRODUCTION

Video surveillance is becoming increasingly important [1]–[4] and can be used as a cyber-security measure to generate early warnings for a variety of problems. An example of an application domain is a video-based fire detection surveillance system used to detect events of fire or smoke to call for assistance, or notify the proper authority and control the social and ecological ramifications [5].

Fire detection has been explored extensively in the computer vision literature. The fast-paced growth of fire detection algorithms demonstrates the continued importance of computers to enable timely rescue measures [6]. Early research focused on fire detection expanded from color detection [7]–[9], moving object detection [10]–[13], and Fourier-based motion and flicker analysis [14]. Wavelet based methods attracted the attention of researchers, as they could easily analyze the color variation in fire and smokes [7], [15], [16]. Another series of approaches are dynamic texture analysis, in which, textures with motions are detected using specific metrics like boundary roughness/disorder analysis [17], [18]. These methods rely on histogram analysis, and are often regarded as rule-based methods. Hybrid approaches have also been developed, combining color and spatio-temporal methods together [19], [20]. In this approach, a tensor of time-height-width is used to impose a covariance descriptor over image regions that change abruptly through time [19]. The descriptor replaces pairwise multiplication in typical covariance computation with a specific summation method that not only reduces the computation costs, but also sums the covariance results over the time frames [19]. In certain cases, smoke or fire detection can be framed as an anomaly detection problem for which a variety of deep learning techniques have been explored [21]. Finally, around 2010 classification based techniques raised a lot of interests among researchers. For example, kernel methods, such as Radial Basis Functions (RBF) have been used to extract nonlinear features for Support Vector Machines (SVM) and Adaboost classifiers [22], [23]. Other classifiers such as Bayesian models [16], [24], Markov based detectors [7], [25], and more recently deep learning models [26], [27] are applied to classification tasks and are frequently referenced in highly-cited publications.

In this paper, we develop and compare several Deep Neural Network models for fire and smoke detection and present a variety of training and validation strategies for high performance in a variety of contexts. We map the fire and smoke detection problem as two sub-problems based on the datasets we explored: (1) a simple fire image classification problem, and (2) a fire and smoke detection and localization problem. For the first sub-problem, two different deep learning models were tested: a basic Convolutional Neural Network (CNN) and
a deep convolutional residual network (ResNet) using a pre-trained ResNet50. For the second sub-problem three models were developed, a Faster R-CNN, an EfficientDet network, and a YoloV5 network. To train and validate the models, a database was compiled from preexisting fire and smoke datasets and from web scraped images with and without fire and smoke. Besides hazardous event detection by fire and smoke detection, the contribution of the paper also includes a dataset compiled from images and videos containing fire and smoke. Images for our dataset were collected with the following considerations:

- The dataset should contain a variety of different contexts such as city, forest, suburban, and kitchen;
- Positive samples should have smoke, fire, or both;
- Images may be taken from a variety of sources such as stock photos and CCTV datasets, which were variably noisy.

The rest of the paper is organized as follows. Section II describes the related work. Methods for collecting, labeling and processing fire and smoke data are described in Section III. Section IV describes the fire image classification models, a basic CNN and a ResNet, and their results. Section V presents the models and results for fire and smoke detection and localization namely a Faster-R-CNN, an EfficientDet and a YoloV5 model. Finally, Section VI and VII present a conclusion and exploration of future work respectively.

II. RELATED WORK

We address the problem of hazardous event detection specifically fire and smoke detection in this study. Therefore, we start by presenting the related work on this problem and then discuss the models that we implemented to provide a background and the reasoning for selecting these models.

A. Fire and Smoke Detection

Celik et al. proposed a method to detect fire in video sequences which compares information about foreground objects with statistical color information of fire [28]. Their statistical fire color model consists of three rules: 1. The value of the red component of an RGB pixel must be greater than the mean of red components of the entire image. 2. The value of the red component of a pixel must be greater than the green component which must be greater than the blue component. 3. Assess and use the ratio of red, blue and green components to filter fire regions. However, the method failed in the case where smoke and no red-colored pixels were present. Also, in a subsequent work on fire detection, Celik et al. used image processing approach to propose a novel model for detection of fire and smoke where a few rules were identified for fire pixels and then given to a Fuzzy Inference System (FIS) in the RGB and YCbCr color spaces [29]. Then, a rule table was formed depending on which pixel was found to be fire with a probability value. The model had 99% accuracy but it could not be used for real-time monitoring.

B. Faster R-CNN

Among the most recent deep learning studies on fire and smoke detection, Faster R-CNN proved to be reliable and precise [30]. Faster R-CNN is efficient in extracting information from both regions of interest and image texture simultaneously through layer sharing [31]. It uses a CNN to create an image feature map, and then uses that feature map to find region of interest proposals for bounding boxes, and pools the image features in each proposed region to classify the same.

Barmpoutis et al. proposed a novel image-based fire detection approach, using deep learning and multidimensional texture analysis. The authors first created a cloud-like manifold of texture using a linear dynamic model, and then tuned its parameters to match fire sections in the image. They also implemented Faster R-CNN on their data to compare with their proposed model. They concluded that Faster R-CNN and their model competed very closely to each other, with F-scores around 95% [30]. Zhang et al. detected smoke on a dataset of wild-land forest smokes concluding that Faster R-CNN provided better accuracy than other CNNs [32].

C. Yolo

In 2016, Redmon et al. proposed a unified architecture: Yolo (You Only Look Once), which used a single neural network to predict bounding boxes and class probabilities directly from full images in a single stage [33]. In 2017, Redmon et al. proposed a modified version of Yolo, called YoloV2 which applied many improvements such as batch normalization, removed dropout, and added pass-through layers to increase the speed and accuracy [34]. In 2018, Redmon et al. provided a third version of Yolo, YoloV3, which applied many small improvements [35].

In 2020, two significantly improved versions of YoloV3 emerged: YoloV4 and YoloV5. Bochkovskiy et al. proposed YoloV4 which included a bundle of improvements on YoloV3. The new features included Weighted-Residual-Connections (WRC), Cross-Stage-Partial-connections (CSP) which reduced the parameters, Cross mini-Batch Normalization (CmBN), Self-adversarial-training (SAT), Mish activation function, Mosaic data augmentation, DropBlock regularization, and CIoU loss [36]. Jocher et al. released YoloV5 soon after YoloV4 which used complex data augmentation (including scaling, color space adjustments, and mosaic augmentation), focus layer, LeakyReLU activation function, and GIoU loss. YoloV5 is written in PyTorch, making it easy to develop and deploy [37].

Li et al. proposed an image fire detection algorithm based on a variety of CNN models, showing that YoloV3 and Faster R-CNN have the highest accuracy with YoloV3 having at least 5% higher accuracy than the other methods [38].

D. Model Scaling and EfficientDet

In 2016, He et al. developed ResNet50 [39]. The residual network architecture is comprised of weighted layers, in this case convolutional layers, and residual connections. The purpose of the residual connections is to enable the training
of much deeper networks. Since the default behaviour of the network is to pass forward the original signal, there is a much higher probability that important information is not lost over a series of layers. A significant advantage of ResNet50 is that its weights are available, and it performs very well on a number of image recognition tasks. This allows ResNet to be highly reusable.

In order to obtain higher accuracy in image classification, it is common to scale up feature extraction by enlarging CNNs in terms of width, depth and resolution [40]. Scaling network depth is a common way for deeper networks to capture richer and more complex features. For example, ResNet [39] achieves higher accuracy by gradually increasing the number of convolution layers such as ResNet18, ResNet34, ResNet50, ResNet101 and ResNet152. Scaling network width (e.g., MobileNets [41]) tends to capture more fine-grained features and makes networks easier to train. In addition, enlarging input images helps in getting higher accuracy [37], because networks can potentially capture more patterns in higher resolutions.

EfficientDet [40] is build with compound model scaling method and like most object detectors, it consist of three parts: backbone, neck, and head. A backbone such as ResNet [39] is pre-trained on ImageNet and is applied to extract features from the input. Some of the object detectors add a neck such as the Feature Pyramid Network (FPN) [42], Path Aggregation Network (PAN) [43], and bi-directional feature pyramid network (BiFPN) [40]) to collect feature maps from multiple dimensions. The head of an object detector is applied to predict classes and bounding boxes. Object detectors can achieve higher accuracy by using a larger backbone network. The EfficientDet architecture is scaled with a compound coefficient to improve the accuracy and efficiency which is crucial for fire hazards detection [40].

### E. Comparison of Detector Models

Object detectors which use bounding boxes must be compared differently than classifiers since the evaluation needs to take into account whether an object of interest was correctly recognized in addition to how well the candidate bounding box lines up with labeled bounding boxes. A common metric for comparing models on these problems is the average precision (AP) for a given minimum intersection over union (IoU) between the candidate and label bounding boxes. This metric is written as \( AP^{50} \) where 50 is the chosen IoU. The mAP metric used for comparison on the COCO dataset [44] is the mean AP across all classes and for 10 different of IoU thresholds.

Faster R-CNN [31] is a two-stage detector which achieved the highest mAP of 78.8 at 200ms per frame. Both EfficientDet [40] and YoloV5 [45] are one-stage detectors. Comparing the larger versions of these models, EfficientDet-D7x reached the highest mAP of 55.1 at 153ms per frame while YoloV5x reached the mAP of 50.4 at speed 6.1ms per frame. Comparing the smaller versions of these models, YoloV5s reached the highest mAP of 36.7 at 2.0ms per frame while EfficientDet-D0 reached an mAP of 34.6 at 6.1ms per frame. A summary of the models and their performances is shown in Table I.

### III. DATA PREPARATION

#### A. Data Collection

One of the contributions of this study is a dataset containing fire and smoke images and videos that we created by collecting images and videos from multiple sources as presented in Table II.

We collected the data from existing sources, shown in Table II, and created four datasets which we refer to as DS1 to DS4 containing images and video frames. We used both image and video data in developing our models. We created image datasets from video frames as explained below.

DS1 is composed of 1,135 fire images and 1,135 non-fire images of various sizes which were reshaped to 256×256. We used this dataset for image classification.

DS2 image dataset is used for fire detection and localization model development and validation. It is made from the furg-fire-dataset containing videos [46] which includes 23 videos at 30 fps. Every frame of the videos was saved to create 28,022 images of different scenes such as barbecue, car, house, wildfire and fire extinguisher training. Several instance frames from the dataset with ground truth annotations are shown in Fig. 1. Every 20 frames of the videos were saved as validation images and the rest of the frames were saved as training dataset. Therefore, DS2 contains 26,612 and 1,410 training and validation images respectively. The final composition of DS2 is shown in Table III.

DS3 is a hand-labeled fire and smoke detection and localization (FSDL) dataset made of 886 images which were collected from a variety of sources. The original images did not have any label. We manually labelled 709 training and 177 validation images using the bounding box tool described in Section III-B. Some instances of DS3 are shown in Fig. 2.

DS4 is another hand-labeled fire and smoke detection and localization dataset that was made to train YoloV5 since it

<table>
<thead>
<tr>
<th>Model name</th>
<th>Faster R-CNN</th>
<th>EfficientDet-D7x</th>
<th>YoloV5x</th>
<th>EfficientDet-D0</th>
<th>YoloV5s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dataset</td>
<td>COCO</td>
<td>COCO</td>
<td>COCO</td>
<td>COCO</td>
<td>COCO</td>
</tr>
<tr>
<td>mAP</td>
<td>78.8</td>
<td>55.1</td>
<td>50.4</td>
<td>34.6</td>
<td>36.7</td>
</tr>
<tr>
<td>Inference time (ms)</td>
<td>200</td>
<td>153</td>
<td>6.1</td>
<td>10.2</td>
<td>2.0</td>
</tr>
</tbody>
</table>

#### TABLE I: Performance and configuration of different models in object detection.
TABLE II: Data sources used to create the Fire-Smoke dataset.

<table>
<thead>
<tr>
<th>Source</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CCTV [47]</td>
<td>Annotated CCTV fire, non-fire images+videos</td>
</tr>
<tr>
<td>Kaggle [48]</td>
<td>Annotated fire, non-fire images+videos</td>
</tr>
<tr>
<td>Outdoor data</td>
<td>Annotated outdoor fire, non-fire images+videos</td>
</tr>
<tr>
<td>Master-Dataset</td>
<td>Annotated fire, non-fire images+videos</td>
</tr>
<tr>
<td>Furg-Fire [46]</td>
<td>Annotated fire, non-fire images+videos</td>
</tr>
<tr>
<td>Web [51]</td>
<td>Unlabelled fire images from the Web</td>
</tr>
</tbody>
</table>

TABLE III: DS2: Fire Detection and Localization Dataset (furg_fire_dataset).

<table>
<thead>
<tr>
<th>Images</th>
<th>Fire</th>
<th>Non-Fire</th>
<th>Annotations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>26,612</td>
<td>15,671</td>
<td>12,941</td>
</tr>
<tr>
<td>Validation</td>
<td>1,410</td>
<td>724</td>
<td>686</td>
</tr>
</tbody>
</table>

required square images (called FSDL-Yolo). It contains 2,096 images scraped from www.bing.com/images of which 1,729 images were used for training and 367 images for testing. The images were scaled and filled with black blocks to reshape to 1280 × 720 which maintained the aspect ratio of the image. Then the images were labelled in the format as needed by Yolo using the software Labelling [52]. Next, the images were resized to 224 × 224, 320 × 320, and 640 × 640. Three image sizes were created to test different training efficiency and achieve the best accuracy.

B. Image Annotation

To create the fire and smoke datasets (FSDL and FSDL-Yolo) with bounding boxes, we labeled fire and smoke images using a bounding box tool [53]. To label the data the following principles were followed:

- Minimize the total number of boxes, only using multiple boxes when it reduced the number of pixels in the bounding boxes by 50% or more;
- Minimize the amount of pixels in the bounding boxes that did not contain fire or smoke;
- Fire and smoke bounding boxes could overlap.

The first principle was used to ensure consistent labelling and to reduce the number of small boxes. An example of the
TABLE IV: Classification accuracy on our Fire Image Classification Dataset.

<table>
<thead>
<tr>
<th></th>
<th>Training Accuracy</th>
<th>Testing Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline CNN</td>
<td>0.9953</td>
<td>0.8546</td>
</tr>
<tr>
<td>Randomly initialized ResNet, fully trained network</td>
<td>0.9906</td>
<td>0.8767</td>
</tr>
<tr>
<td>ResNet original weights, fully trained network</td>
<td>1.0000</td>
<td>0.9714</td>
</tr>
<tr>
<td>ResNet original weights, head trained network only</td>
<td>0.9741</td>
<td>0.9295</td>
</tr>
</tbody>
</table>

The first principle is shown in Fig. 3b where the blue bounding box contains multiple distinct fires and could be broken into several smaller bounding boxes. However, the total size of the smaller boxes would not be less than 50% of the single box and therefore the single box was used. Fig. 3b is also an example of fire and smoke bounding boxes overlapping. Overlapping boxes can confuse the model as parts of images are labelled as both fire and smoke, and this ambiguity can affect the performance of the network. However, we allowed overlapping labels to reduce the time spent in labelling the data where fire and smoke had a lot of overlap in the data.

C. Annotation Issues

Due to picture quality and the transparency of smoke it was difficult to identify and label fire and smoke in some cases. This is demonstrated in Fig. 3c. This could result in smoke bounding boxes that are too big and contain pixels that do not have smoke, or boxes that are too small and do not contain all of the smoke.

Another issue was labelling images that entirely contain fire or smoke, similar to Fig. 3d. Labelling these images is trivial but these images could affect model performance.

IV. EXPERIMENT I: CLASSIFYING FIRE IMAGES

In this experiment, we compare a basic CNN model with the ResNet50 model trained using different approaches to classify images based on whether or not they contain fire. Based on the literature review, both the above models had good performance for image classification problems.

A. Methodology

1) Basic CNN: The architecture of our basic CNN model for classifying fire images is shown in Fig. 4. The feature extraction part of the model includes two blocks, each block includes two convolution layers and one max pooling layer. The classification part includes two dense layers with final output size equal to 2 for the fire and non-fire classes. The loss function for training is the categorical cross-entropy.

Our baseline fire image classification architecture is implemented using the PyTorch framework. The model is trained and evaluated on Tesla V100 GPU with 32 GB RAM, CUDA v10.1 and cuDNN v9.1., using Adam optimizer, a learning rate of 0.001, weight decay of 0.01, and a batch size of 32.

2) ResNet50: The model architecture used for these experiments consists of 4 main components: a ResNet base model, a global average pooling layer, a dropout layer, and a linear classifier (see Fig. 5). The ResNet model used is ResNet50, using the weights available from tensorflow.keras.applications.resnet50. For all experiments, the final layer of the model is not used which is a common practice since the final values are too specific to the classes in the Imagenet dataset.

The global average pooling layer is used to collapse the image, regardless of its initial dimensions, into a single feature vector. This consistent representation lets us avoid carefully resizing, cropping, and stretching images which could potentially affect the models’ generalization performances. Additionally, since we are training on a relatively small subset of images, dropout (probability=0.2) is used to reduce the impact of overfitting.

The last component is a fully connected layer leading to the two outputs, fire or not fire (normalized with softmax). This layer’s weights are randomly initialized. The loss used for the experiments conducted was categorical cross-entropy so that the model could be generalized to more classes.

We explore the efficacy of using pre-trained ResNet [39] weights when training classifiers on novel fire data. Using existing weights helps limit the required gathering, labelling and storing of fire data. We trained three different versions...
of the network: a) randomly initializing and training all the network’s weights; b) using ResNet initial weights and training the whole network; and c) using ResNet initial weights and training only the last layer of the model.

The whole model trained from scratch is used as a baseline to compare the accuracy achieved by the different training approaches. The other two training strategies indicate how well the weights learned for ImageNet transfer to the problem of fire detection. The network was trained using Adam optimizer with a learning rate of $10^{-4}$ and batch size of 32.

B. Results

The results are shown in Table IV. The training set used 80% of the data and 20% was used for the testing set. The randomly initialized ResNet50 was slightly better than the CNN (87% versus 86%, see Table IV). The best validation accuracy was obtained when reusing the pre-trained ResNet50 weights and retraining the whole network (97%). Retraining only the head (or classifier) portion of ResNet50 gave moderate performance (94%). This shows that reusing ResNet50 pretrained weights is useful, but retraining them on the target task improves the overall performance. Our experiments show that existing weights for image recognition can be used as a solid basis for fire classification.

V. EXPERIMENT II: DETECTING AND LOCALIZING FIRE AND SMOKE

In this experiment we evaluate different deep neural network approaches on fire and smoke detection and localization datasets.

A. Methodology

Three networks were developed to detect and localize fire and smoke within images: Faster R-CNN, EfficientDet, and YoloV5 based on their superior performances in the existing literature in object detection and localization.

1) Faster R-CNN: The architecture of Faster R-CNN is composed of two modules. First, a Region Proposal Network (RPN), implemented using a CNN model, proposes the bounding rectangles (so-called regions). Second, a Fast R-CNN detector is applied to the proposed region to classify the contents of the bounded regions.

An RPN learns how to associate feature maps to rectangular object boundaries. It uses rectangular boundaries to propose whether the content of an area can be recognized as an object. The higher the overlap of proposed boundary with the training label boundaries, the higher the objectness score. The score is computed by a regression function. The shared layer and RPN convolution layers are optimized to improve the score. The model slides a small network over the convolutional feature map output to generate region proposals from the shared convolutional layer. At each sliding window location, the feature maps are projected to convolutional kernels (slider) to both classify to the center of labeled bounding box and to classify the sliding window’s width/height to the width and height of label’s bounding box.

The resulting region proposals are fed into the classifier layers to detect the objects. The classifier typically consists of 2 densely connected layers and a number of convolution layers (VGG-16 uses 13 convolutional layers).

We implemented Faster R-CNN and applied the prediction results as the baseline. Then, data augmentation was implemented to increase the variation of input through rotations, scaling, and shearing. Finally the images and labels were used in a Faster R-CNN model with shared layers of VGG16 model [54].

2) EfficientDet: We also evaluated the deep learning object detector called EfficientDet [40] to identify and localize fire in different scenes. Commonly, object detectors consist of three parts: backbone, neck, and head. EfficientDet uses an EfficientNet backbone and a BiFPN neck. The head of an object detector is applied to predict classes and generate bounding boxes. EfficientDet scales the input image resolution, backbone network, BiFPN and number of box/class layers together with a compound coefficient.

Our fire detection architecture is implemented with PyTorch framework, trained and evaluated on Tesla V100 GPU with 32 GB RAM, CUDA v10.1 and cuDNN v9.1, using Adam optimizer with a learning rate of 0.0001 and a weight decay of 0.01. The batch size is 4. The code is referenced from [55]. We augmented the data by flipping images vertically.

3) YoloV5: For detection of fire events in videos, real-time performance is the most challenging part. YoloV4 [36] and YoloV5 [37] are two of the fastest object detection models. YoloV4 provides superior accuracy and YoloV5 is faster. YoloV5 is implemented in PyTorch and has a much smaller model storage size (only 27MB for YoloV5 compared to 163MB for YoloV4). Also, it is easier to develop and deploy in the final product. For these reasons, we tested a solution based on the YoloV5 model.

YoloV5’s PyTorch framework allows half the floating point precision (16 bit precision) in training and inference, it can be used to speed up the inference stage. YoloV5 provided 4 models (s,m,l,x), from small to extra large. YoloV5s is the smallest and the fastest one. The architecture of YoloV5 is shown in Fig. 6.
### TABLE VI: Results of EfficientDet on the furg_fire_dataset.

<table>
<thead>
<tr>
<th>Model</th>
<th>AP</th>
<th>AP95</th>
<th>AP75</th>
<th>FPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>EfficientDet-D0</td>
<td>73.1</td>
<td>91.9</td>
<td>84.4</td>
<td>28.5</td>
</tr>
<tr>
<td>EfficientDet-D1</td>
<td>73.2</td>
<td>93.0</td>
<td>87.0</td>
<td>23.0</td>
</tr>
<tr>
<td>EfficientDet-D2</td>
<td>76.4</td>
<td>93.0</td>
<td>87.8</td>
<td>20.1</td>
</tr>
<tr>
<td>EfficientDet-D3</td>
<td>78.3</td>
<td>94.0</td>
<td>89.2</td>
<td>17.1</td>
</tr>
</tbody>
</table>

Table VI. We stopped increasing the number of compound coefficients of EfficientDet to enlarge the model for getting higher AP until it started increasing the inference time without significant improvements of the AP. EfficientDet achieved an AP of 78.3 and could detect fire at 17 FPS. We did not apply Faster R-CNN on the more complex fire and smoke detection dataset because of the lower AP and detection time as compared to EfficientDet.

### TABLE VII: Results of EfficientDet on the Fire and Smoke Detection dataset (FSDL).

<table>
<thead>
<tr>
<th>Model</th>
<th>AP</th>
<th>AP95</th>
<th>AP75</th>
<th>FPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>EfficientDet-D0</td>
<td>32.3</td>
<td>59.6</td>
<td>31.2</td>
<td>28.5</td>
</tr>
<tr>
<td>EfficientDet-D1</td>
<td>33.3</td>
<td>66.4</td>
<td>28.7</td>
<td>23.0</td>
</tr>
<tr>
<td>EfficientDet-D2</td>
<td>30.4</td>
<td>64.8</td>
<td>28.2</td>
<td>20.1</td>
</tr>
<tr>
<td>EfficientDet-D3</td>
<td>26.2</td>
<td>49.0</td>
<td>24.7</td>
<td>17.1</td>
</tr>
</tbody>
</table>

2) **Fire and Smoke Detection and Localization (FSDL):**

This is the hand-labeled fire and smoke detection an localization dataset with 886 images.

**EfficientDet**: The results of fire and smoke detection on the FSDL dataset are shown in Table VII. Although the fire and smoke detection model could not achieve as high AP as the fire detection model, it still successfully localized fire and smoke in some simple scenes as shown in Fig. 8. However, we only trained EfficientDet with 4 compound coefficients. From the results in Table VII we can see, after EfficientDet-D1, as the compound coefficient increases, the AP decreases, because the dataset is too small and a large model starts overfitting easily. The best results were obtained with EfficientDet-D1 on AP50.

**EfficientDet-D5**: The model is evaluated on the validation dataset and the fire detection evaluation results are shown in Table VI. We stopped increasing the number of compound coefficients of EfficientDet to enlarge the model for getting higher AP until it started increasing the inference time without significant improvements of the AP. EfficientDet achieved an AP of 78.3 and could detect fire at 17 FPS. We did not apply Faster R-CNN on the more complex fire and smoke detection dataset because of the lower AP and detection time as compared to EfficientDet.

### TABLE VII: Results of EfficientDet on the Fire and Smoke Detection and Localization dataset (FSDL).

<table>
<thead>
<tr>
<th>Model</th>
<th>AP</th>
<th>AP95</th>
<th>AP75</th>
<th>FPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>EfficientDet-D0</td>
<td>32.3</td>
<td>59.6</td>
<td>31.2</td>
<td>28.5</td>
</tr>
<tr>
<td>EfficientDet-D1</td>
<td>33.3</td>
<td>66.4</td>
<td>28.7</td>
<td>23.0</td>
</tr>
<tr>
<td>EfficientDet-D2</td>
<td>30.4</td>
<td>64.8</td>
<td>28.2</td>
<td>20.1</td>
</tr>
<tr>
<td>EfficientDet-D3</td>
<td>26.2</td>
<td>49.0</td>
<td>24.7</td>
<td>17.1</td>
</tr>
</tbody>
</table>

3) **Fire and Smoke Detection and Localization (FSDL-Yolo):** To compare EfficientDet to YoloV5, we trained and tested both architecture on our (FSDL-Yolo) dataset. The comparison results are shown in Table VIII.

**YoloV5**: A comparison between the combinations of image size and YoloV5 model size was made. Three image sizes were considered: 224×224, 320×320, and 640×640 for YoloV5s.

---

**Fig. 7**: Fire bounding box by Faster R-CNN.

**Fig. 8**: Fire and smoke detection examples with EfficientDet.
and YoloV5m models. A 100 training epochs were used. The comparison is shown in Table VIII.

From Table VIII, it can be seen that the accuracy of YoloV5m using image size 640×640 performed the best achieving an AP of 46.6 and an AP50 of 77.7. However, the detection time of YoloV5m using the image size of 640×640 was around 4 times that of YoloV5m using image size of 320×320 which achieved an AP of 45.5 and an AP50 of 74.7. Considering the tradeoff between detection speed and accuracy, YoloV5m with image size 320×320 was selected as the solution of YoloV5 and the weights with the best mAP after training was used for the detection model.

The precision, recall, and mAP of YoloV5m with image size 320×320 after 100 epochs of training on the dataset is shown in Fig. 9.

TABLE VIII: Fire and Smoke Detection Evaluation Results with EfficientDet and YoloV5 on FSDL-Yolo Dataset.

<table>
<thead>
<tr>
<th>Model</th>
<th>AP</th>
<th>AP50</th>
</tr>
</thead>
<tbody>
<tr>
<td>EfficientDet-D0</td>
<td>21.9</td>
<td>52.3</td>
</tr>
<tr>
<td>EfficientDet-D1</td>
<td>24.0</td>
<td>56.5</td>
</tr>
<tr>
<td>EfficientDet-D2</td>
<td>26.8</td>
<td>57.7</td>
</tr>
<tr>
<td>EfficientDet-D3</td>
<td>27.9</td>
<td>59.0</td>
</tr>
<tr>
<td>EfficientDet-D4</td>
<td>31.2</td>
<td>62.5</td>
</tr>
<tr>
<td>EfficientDet-D5</td>
<td>28.2</td>
<td>56.7</td>
</tr>
<tr>
<td>YoloV5s (224×224)</td>
<td>40.6</td>
<td>70.2</td>
</tr>
<tr>
<td>YoloV5s (320×320)</td>
<td>43.3</td>
<td>73.5</td>
</tr>
<tr>
<td>YoloV5s (640×640)</td>
<td>40.6</td>
<td>72.4</td>
</tr>
<tr>
<td>YoloV5m (224×224)</td>
<td>43.5</td>
<td>72.9</td>
</tr>
<tr>
<td>YoloV5m (320×320)</td>
<td>45.5</td>
<td>74.5</td>
</tr>
<tr>
<td>YoloV5m (640×640)</td>
<td>46.6</td>
<td>77.7</td>
</tr>
</tbody>
</table>

Applying the trained YoloV5m model on an image of size 320×320, we obtained the result as shown in Fig. 10.

The best performance on the smoke and fire problem was obtained by YoloV5, beating EfficientDet by a significant margin.

Fig. 9: Performance metrics for the training of YoloV5m using igages of size 320×320.

VI. CONCLUSION

Automated image and video based fire and smoke hazard detection can reduce fire related casualties. In this study, we created a fire and smoke image and video dataset by collecting data from multiple sources. We manually labelled a part of this data to train deep learning models for two tasks. Experiment I presents the first task of fire image classification. Experiment 2 presents the second task of fire and smoke detection and localization. Based on the pre-trained weight experiment results using ResNet50, we conclude that utilizing pre-trained weights offers reliable and substantial improvements for fire classification when training data or resources are limited. The Faster R-CNN model results demonstrated that the classification accuracy is not satisfactory. However, the RPN could locate the bounding box patterns well and the classifier accuracy for bounding boxed images is 89%. Our EfficientDet object detection architecture achieved over 70 AP in fire-only detection. However, due to the limitation of our fire and smoke detection dataset, the model only achieves around 30 AP in fire and smoke detection providing good performances despite low resolution. Experiments using YoloV5 showed that the YoloV5m model with image size 320×320 is one effective candidate model for fire and smoke detection.

Fig. 10: Fire and smoke detection examples with YoloV5m.
detection in a video stream.

We can conclude that detecting smoke is more difficult than fire. This issue is reflected in the accuracy of the EfficientDet model, as the EfficientDet model performance decreased when detecting fire and smoke as opposed to just fire. This may stem from the difficulties in visually identifying smoke, and as a consequence, it is difficult to create high quality bounding box labels for smoke which negatively affects model performance.

VII. Future Work

The models evaluated in this project are among the top Object Detectors in the Computer Vision literature. While the current paper brushes a good picture of the performances of each architecture, many strategies could be used to further improve the accuracy of our models. In particular, image preprocessing methods could be explored, the models could be trained on larger sets of labeled images, and data augmentation could be used to improve our results. Labeling a large dataset manually has been shown to be particularly difficult and certainly a key to develop better models. Implementations of accurate fire image classification or fire and smoke detection and localization models could improve early fire hazard detection rates or help track the progression of forest fires.

Data preprocessing methods for fire images which have shown promising results even without using deep learning include time-frequency domain analysis methods such as wavelet-entropy, cosine transform, differential texture analysis, and pyramid matching [30]. These methods deal with the texture of fire segments that mainly resides in the frequency domain rather than the spatial domain. Extraction of spectral information facilitates more filter-ready processing in the learning phase. With spectral information, one can also delve deeper in space-frequency interrelations through spectral deep learning models like Graph Convolutional Network (GCN) and CNN.

Another strategy to explore would be to filter relevant fire information using multi-resolution analysis. Multi-resolution analysers are a cascade of detectors and classifiers that zoom in at image from big picture (globality) to details (locality) to more accurately search for fire and smokes [56].

Beyond the above mentioned work, the following approaches could also improve the performance of fire and smoke detectors:

- Using caption analysis to restrict video frames on fires;
- Combining caption and frame analysis in shared task learning;
- Using Graph Convolutional Network and other spectral analysis tools for focusing on the texture itself;
- Expand labelled image database and evenly distribute the type of image (city, forest etc.) in the database;
- Image semantic segmentation is pixel-wise prediction which can classify each pixel into its category and help understand the images at the pixel-level. It can separate fire and smoke more accurately in the images or video frames.

REFERENCES


**DOCUMENT CONTROL DATA**

1. ORIGINATOR (Name and address of the organization preparing the document. A DRDC Centre sponsoring a contractor’s report, or tasking agency, is entered in Section 8.)
   
   Institute of Electrical and Electronics Engineers (IEEE)
   3 Park Avenue, 17th Floor
   New York, NY 10016-5997 USA
   ieee-mce@ieee.org

2a. SECURITY MARKING (Overall security marking of the document including special supplemental markings if applicable.)
   CAN UNCLASSIFIED

2b. CONTROLLED GOODS
   NON-CONTROLLED GOODS
   DMC A

3. TITLE (The document title and sub-title as indicated on the title page.)
   FireWarn: Fire Hazards Detection Using Deep Learning Models

4. AUTHORS (Last name, followed by initials – ranks, titles, etc., not to be used)
   Hogan, I.; Qiao, D.; Luo, R.; Moattari, M.; Carthy, A.; Zulkernine, F.; Rivest, F.; Breton, M.

5. DATE OF PUBLICATION (Month and year of publication of document.)
   December 2021

6a. NO. OF PAGES (Total pages, including Annexes, excluding DCD, covering and verso pages.)
   10

6b. NO. OF REFS (Total references cited.)
   56

7. DOCUMENT CATEGORY (e.g., Scientific Report, Contract Report, Scientific Letter.)
   External Literature (P)

8. SPONSORING CENTRE (The name and address of the department project office or laboratory sponsoring the research and development.)
   DRDC – Valcartier Research Centre
   Defence Research and Development Canada
   2459 route de la Bravoure
   Québec (Québec) G3J 1X5
   Canada

9a. PROJECT OR GRANT NO. (If appropriate, the applicable research and development project or grant number under which the document was written. Please specify whether project or grant.)
   02cb

9b. CONTRACT NO. (If appropriate, the applicable number under which the document was written.)

10a. DRDC PUBLICATION NUMBER (The official document number by which the document is identified by the originating activity. This number must be unique to this document.)
   DRDC-RDDC-2022-P002

10b. OTHER DOCUMENT NO(s). (Any other numbers which may be assigned this document either by the originator or by the sponsor.)

11a. FUTURE DISTRIBUTION WITHIN CANADA (Approval for further dissemination of the document. Security classification must also be considered.)
   Public release

11b. FUTURE DISTRIBUTION OUTSIDE CANADA (Approval for further dissemination of the document. Security classification must also be considered.)
13. ABSTRACT/RÉSUMÉ (When available in the document, the French version of the abstract must be included here.)

Hazardous situations such as house, car, or forest fires may be recorded by cameras long before they are identified by people. To test whether deep learning could be used to quickly detect fires, we performed a series of experiments to detect the presence of fire or smoke in images and labeled them with bounding boxes. Two custom datasets were created in this research: a fire image classification dataset, and a fire and smoke detection and localization dataset. The first one only classifies the whole image, while the detection set further provides information about where the fire or smoke is within the image. We explore the efficacy of a basic convolutional classification neural network, which proved effective for fire classification, but show that pretrained classification models such as ResNet improves the accuracy when classifying fire and non-fire images. The pretrained model achieves 97.14% testing accuracy on our fire classification dataset. For fire detection and localization, three models were trained on images of fire and smoke to find and label the regions of interest. Results show that Faster R-CNN did not perform very well on fire detection and localization, while EfficientDet and YoloV5 performed much better. Moreover, YoloV5 using low resolution images also performed well on smoke detection and localization, which is more difficult than fire. YoloV5 achieved an average precision of 46.6 on our fire and smoke detection dataset.