# Forest Fire Detection Using a Hybrid Convolutional Neural Network and Support Vector Machine Approach

**Charlie Gallop**
MSc. Applied Data Science
University of Exeter

**Meg Lyons**
MSc. Applied Data Science
University of Exeter

*Forest fires are responsible for substantial damage to ecosystems, infrastructure, and human health annually. Therefore, early and reliable detection is essential, to ensure that damage can be avoided or limited. While there exists many traditional methods for fire detection, these methods are often expensive, unreliable, or impractical. As such, an updated method is needed and in recent years, there has been a focus on using convolutional neural networks (CNN) to help tackle this problem. Real-world fire detection, however, poses the challenge of using technology in remote environments with limited technological infrastructure. Therefore, this research builds on the existing knowledge of the effectiveness of CNNs for early fire detection and aims to compare the performance of several light-weight state-of-the-art (SOTA) CNN models by way of transfer learning. We then attempt to improve the performance of these models by using image augmentation, and a support vector machine (SVM) at the classification layer to optimize for risk aversion. The models are tested on two datasets, to test for generalisability and evaluate performance on a variety of images. The models are trained on dataset 1, containing cleaner images, with no humans or human infrastructure, than dataset 2. They are then tested on both, and results indicate that using SOTA models can be effective for early fire detection. The performance was further improved by addition of the SVM classification layer.*

## 1 Introduction

Fast and accurate forest fire detection is essential to prevent social, ecological, and economic disasters [1]. Forest fires can be natural occurrences that help maintain health within some ecosystems containing pyrophytic plants, by aiding seed dispersal and releasing nutrients in the soil [2]. However, human impact on the natural environment is causing more frequent, higher intensity and more widespread fires [3].

Forest degradation and fragmentation is leading to more fire prone areas, and anthropogenic climate change is causing droughts and extreme temperatures. Moreover poor fire policy, which has encouraged prematurely suppressing wildfires, has lead to decades of fuel accumulation that is accelerating the destruction. This build up of fuel must be reduced so that when fires do occur, they do not move through the tops of the trees independently but burn along the surface at a lower intensity. Therefore consuming smaller trees and restoring forest resilience. This can be done through prescribed fires, intentionally setting controlled fires, or opportunistically, by properly managing wildfires [4] through early detection methods. Environmental tipping points, such as permafrost thawing in the Arctic, can also been triggered by wildfires which would have irreversible and abrupt climatic effects [5]. Furthermore, the combination of direct carbon dioxide released and the loss of carbon sequestration potential will, in turn lead to more fire prone conditions [6].

There are a myriad of negative consequences from uncontrolled forest fires including prompting public health crises from the air pollution, loss of life and crippling of local economies [7] [1]. For example, the 2019/2020 fire season in Australia is known as the 'Black Summer' due to the devastating effects of the wildfires caused by record breaking heat and low rainfall [8] [9]. Directly and indirectly, hundreds of people lost their lives, thousands of people were hospitalised and over a billion animals were killed [10] [11]. The fires burnt over 19 million hectares of of land, and caused billions of dollars impact to the economy; highlighting the importance of early detection of such fires [12] [13]. Fires can become uncontrollable, and thus extremely damaging, very quickly. Therefore, detection frameworks should be trained to be risk-averse, to minimize type II errors.

A variety of forest fire detection methods are currently in use globally. These methods range from human-based observation techniques, such as in-situ fire-watchers and apps or phone lines that allow the public to report any sightings of fire or smoke, to the more advanced multispectral imaging sensors that detect signs of forest-fires autonomously. Due to increased technological advancements in digital imaging, optical image sensors are the most commonly used fire detection method [14] [15]. A review by Alkhatib found that the most popular software used alongside optical sensors are EYEfi, ForestWatch, and FireWatch, with all of them still being deployed in several locations globally. Despite this, these

systems are unreliable and have high false alarm rates with a trained human observers outperforming the vision based sensors [15].

Traditional software and systems used for early fire detection require engineers to manually undertake feature extraction which requires a high level of competency and expertise to be able to design an effective system [16]. As such, these systems are often expensive and time consuming to implement. This combined with the high false positive rate [15] of these systems creates an opportunity for deep-learning to be used to help combat these issues and build upon the infrastructure used for traditional detection systems to deploy improved models for early fire detection.

Deep learning based fire detection systems offer an improvement over traditional methods since they automatically extract the relevant features needed for fire identification. Furthermore they can be trained using a variety of different images helping to limit the amount of false alarms caused by things like fallen trees, clouds and shadows where these anomolies would often trigger traditional systems [15]. Another challenge of early fire detection is that they often occur in remote areas with highly limited to no technological infrastructure making it difficult to deploy complex sensing devices that require large amounts of energy or mobile phone signal. As such there has been a focus on designing and using light-weight neural networks that are designed for use on mobile devices and are energy efficient whilst providing the accuracy needed to detect fires.


Fig. 1: Nine example images from and their associated labels dataset 1.

## 2 Current work

### 2.1 Neural Networks in Forest Fire Detection

Previous work has used various neural network architectures (long short-term memory (LSTM), gated recurrent unit (GRU), simple recursive neural networks (RNN)) with a high degree of accuracy when used on drone systems [17], however convolutional neural networks (CNN) have been most commonly used for fire-detection models on ground based devices [18]. Generally, the models used in research utilize transfer learning (TL) and modify existing CNN architectures such as VGG-Net [19], GoogleLeNet [20], AlexNet [21], and ResNet [22], to achieve a high level of accuracy [18]. While these models offer good performance, they can require substantial computing power to train and deploy due to the large amount of parameters included in the models, making them not very energy efficient for use in forest fire detection scenarios where energy infrastructure may be limited. As such there has been a focus on the use of light-weight CNNs (LW-CNN) for use in forest fire detection. Yar et al. designed a new LW-CNN for real-time forest fire detection and compared it to transfer learning models (AlexNet [21], VGG16 [19], ResNet50 [22], MobileNetV1 [23], NASAetMobile [24]) to test it's performance. The model created by the authors had less parameters than the other models and outperformed them in both the false positive rate and accuracy metrics [16]. Similarly, Muhammad et al. designed a LW-CNN model based on MobileNetV2 for fire-detection [25]. There have been improvements made in the models available for use in TL, particularly models designed for mobile devices, and hence this research focuses on comparing these newer models to asses their ability to detect forest fires.

### 2.2 Hybrid CNN-SVM

In recent years, there has been some research into hybrid approaches to classification problems that are achieving high levels of accuracy [26]–[28]. These utilize CNNs for automatic feature extraction, and then feed the layer into a classification algorithm. The support vector machine (SVM) algorithm is structured in such a way that allows for risk minimisation, and therefore is an appropriate classifier for fire detection [29]–[32].

The proposed methodology will therefore utilize TL to train three LW-CNNs for binary classification: MobileNetV3 [33], EfficientNetB0 [34], and EfficientNetB4 [34] (Table 1). These models will then be modified to ensure that the algorithm is risk-averse. An SVM at the classification layer will be used for this, optimising for the recall metric, i.e. minimising false negatives.

## 3 Methods

### 3.1 Data

The data used to train these models was curated by Khan and Hassan (2020) and sourced from multiple search engines [35]. There are 1900 images in total: all 3-channelled RGB and each with a spatial resolution of 250 x 250 pixels. It is a balanced dataset, with a 80:20 training/testing split. This dataset will be referred to as dataset 1 and an example subset can be seen in Figure 1.

As the dataset is relatively small, image augmentation will also be used. This is a pre-processing technique that

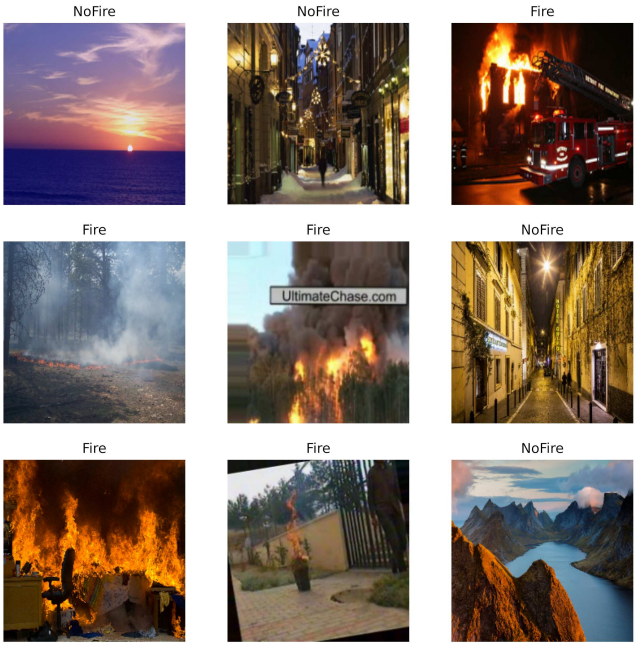


Fig. 2: Nine example images from and their associated labels dataset 2.

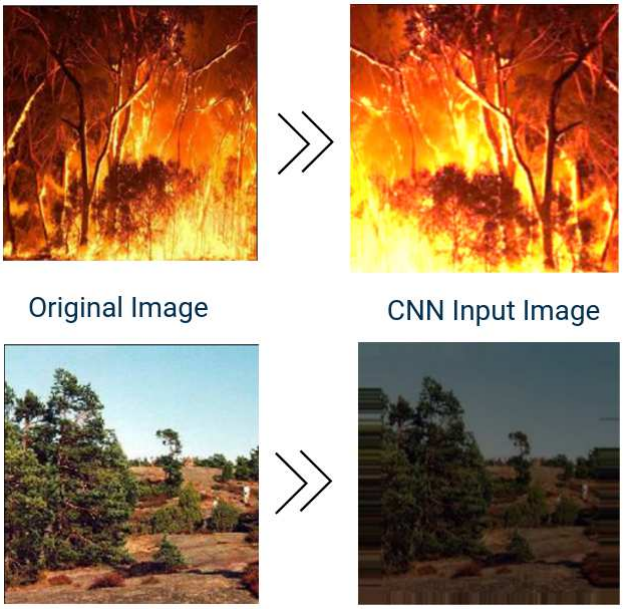


Fig. 3: Example of the augmentations that ImageDataGenerator applies to the images before being fed into the CNN.

transforms existing images whilst preserving their label, and can improve model performance by reducing over-fitting and improving accuracy [36]. The `ImageDataGenerator` class from `keras` will be utilized for this [37]. Real-time transformations of images, chosen randomly from a specified argument list, are created, and fed into the CNN (Figure 3). The possible augmentations include geometric transformations and contrast enhancement (Table 8 in appendix). This ensures that the model receives a slight variation of each image at each epoch, thus aiming to improve robustness. As augmentation is not a guarantee of model improvement, the models will be first tested without any augmentation.

The trained models will also be tested on a second dataset, dataset 2 (Figure 2), to check for generalisability. This is also a balanced dataset, with 2204 images in total and was assembled by Jadon et al [38].

### 3.2 Model Selection

A CNN architecture was selected for this project due to it's suitability for image classification problems [39][40]; the end goal of the model is to be able to determine if an image contains fire or does not contain fire and CNNs provide an efficient and accurate solution to this problem. Transfer learning is the process of leveraging previous knowledge gained from tackling previous problems and using it to more efficiently tackle new problems [41]. Therefore, since there exists a selection of CNN models that have been designed by highly skilled engineers to tackle classification problems, we chose to utilise transfer learning for this project to assess the performance of these state-of-the-art models on forest fire detection. These models can come in a variety of sizes with varying performance (see Figure 4). As such, for this research, models with a smaller size (less parameters) that still achieved a relatively high top 1 accuracy were selected for comparison. In addition, there was a focus on selecting models that have specifically been designed for use on mobile or low-power devices. Using these criteria, MobileNetV3_small, EfficientNetB0, and EfficientNetB4 were selected for comparison (Table 1).

The MobileNetV3 models were designed and optimised for use on mobile devices and improvements have been made in performance compared to the previous MobileNetV2 versions. In addition, MobileNetV3_small was selected since it was specifically designed for low-resource use cases. For image classification problems, MobileNetV3_small has improved by 6.6% in comparison to a similar MobileNetV2 model [33]. The performance of MobileNet models for forest-fire detection can be seen in the results achieved by Yar et al., where MobileNetV1 achieved a competitive recall and F1-score when compared to larger models such as ResNet50 and VGG16 [16].

The EfficientNet architecture uses model scaling of MobileNet and ResNet models by using a novel scaling method to increase the accuracy and efficiency of the models. This results in a larger model with more parameters, so the smallest EfficientNet model, EfficientNetB0, was included in the comparison to compare against MobileNetV3_small. EfficientNetB4 was also included due to this having best trade-off between parameter size and model performance while still being relatively small in size [34].

### 3.3 Training the selected models

The `keras.applications` API was used to download the models and import the pre-trained weights. When importing the models, the classification layers were not included and

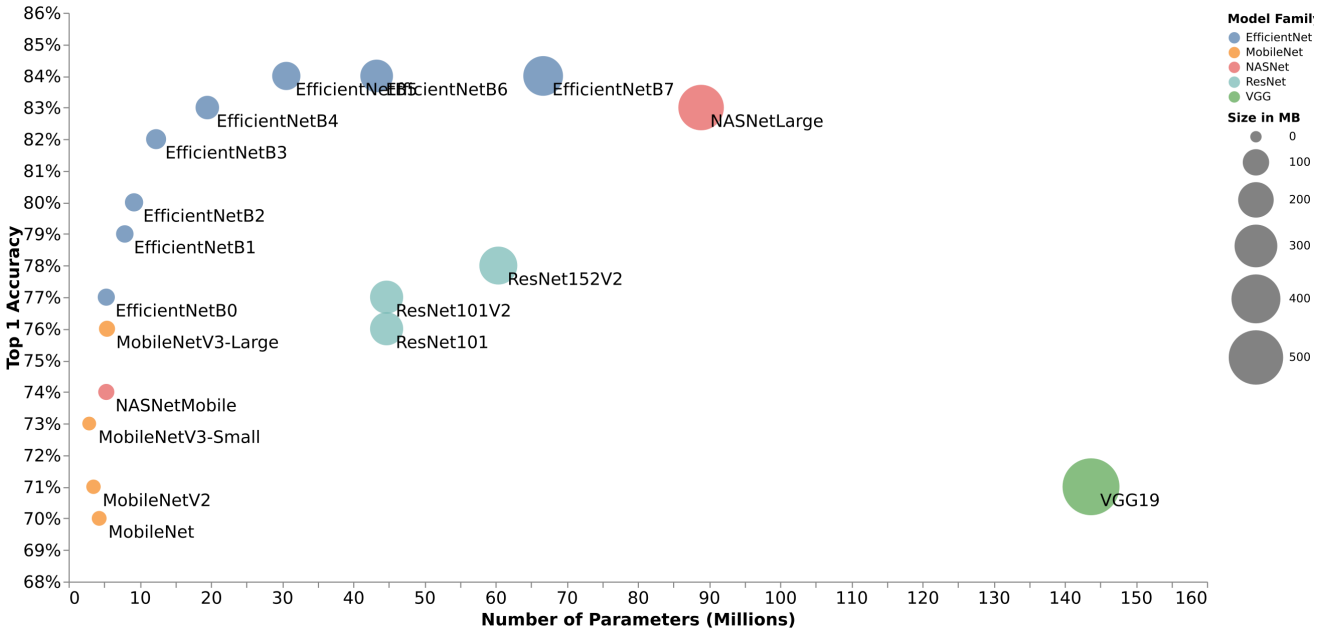


Fig. 4: Top 1 accuracy vs the amount of parameters in the model. The size of the points represent the size of the models in Megabytes (MB). Data sourced from Keras Applications API [42].

Table 1: Comparison of selected models

| Architecture | Parameters (millions) | Top-1 Accuracy | Size (MB) |
|---|---|---|---|
| MobileNetV3_small | 2.9 | 73.1% | 10.9 |
| EfficientNetB0 | 5.3 | 77.1% | 29.0 |
| EfficientNetB4 | 19.5 | 82.9% | 75.0 |

a custom classification layer, consisting of an average pooling layer and a dense layer with sigmoid activation, was added to the base model. Initially the models were trained on dataset 1 using the pre-trained weights associated with the models until they converged. Fine tuning, a method of making a proportion of the weights trainable, was then used to gain incremental improvements in the performance of the models. The performance of these models as well as the trained weights were saved. This process was then repeated on augmented data and again these models with the trained weights were saved. Figure 5 summarises the flow of this project. Each model was trained on an AMD RYZEN 7 5800H CPU with 16 Gigabyte (GB) onboard ram.

### 3.4 Support Vector Machines

Pioneered by Valadimir Vapnik [44], SVMs are a popular supervised machine learning algorithm due to their impressive empirical performance [45]. They are predominantly used for binary classification problems (although can also be extended to multi-class classification, regression and outlier detection) and are non-probabilistic. They are memory efficient and particularly useful for small datasets, non-linear data or when the number of parameters exceeds the sample size.

Utilising computational learning theory, the optimal hyperplane to classify data points is found by maximising the margin. In it's simplest form, when the data is linearly separable and a hard margin is possible (Figure 6), there exists a hyperplane

$$\vec{\omega}^T \vec{x} + b = 0, \tag{1}$$

that separates the two classes such that

$$\begin{aligned} \vec{\omega}^T \vec{x}_i + b \geq 0, \text{ for } y_i = +1, \\ \vec{\omega}^T \vec{x}_i + b \leq 0, \text{ for } y_i = -1, \end{aligned} \tag{2}$$

and the decision function

$$g(\vec{x}) = \vec{\omega}^T \vec{x} + b \tag{3}$$

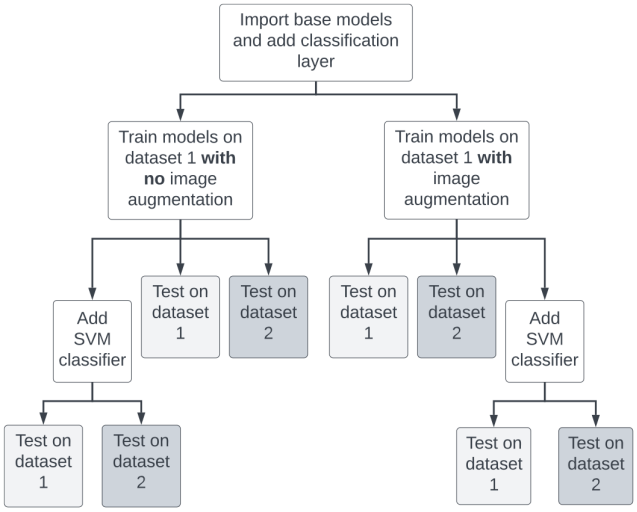


Fig. 5: The process used for training and testing the models.

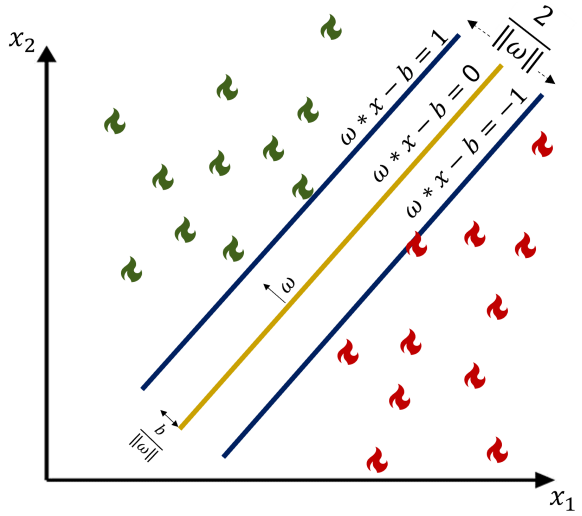


Fig. 6: Maximum-margin hyperplane and margins for binary SVM classifier. Datapoints touching the margin are called the support vectors. Adapted from [43].

can therefore be thought of as the functional distance from the hyperplane. There are however two types of margins in SVMs: hard and soft. When the margin is hard, there are no misclassifications allowed whilst training however this can lead to over-fitting, and may not always be possible. A soft margin allows for misclassfications whilst training, and can be more generalisable. The hinge loss function is utilised for this:

$$max(0, 1 - y_i(\mathbf{w}^T \mathbf{x}_i - b)), \tag{4}$$

and

$$C||\mathbf{w}||^2 + \left[\frac{1}{n}\sum_{i=1}^{n} max(0, 1 - y_i(\mathbf{w}^T \mathbf{x}_i - b))\right], \tag{5}$$

is now minimised, where C is the regularisation parameter. As C increases, the margin of the hyperplane gets smaller. Moreover, another reason why SVMs are a popular technique is the kernel trick. Kernels allows a linear classifier to be used on a non-linear problem by mapping data to a higher dimension. Explicitly applying these transforms would have high and impractical computational costs, however the kernel trick allows operations to occur in an implicit feature space [46]. The kernels that will be utilized for this project are the radial basis function (RBF):

$$\exp(-\gamma||\vec{x_i} - \vec{x_j}||^2) \text{ for } \gamma > 0 \tag{6}$$

and sigmoid:

$$\begin{gathered}\tanh(\gamma \vec{x_i} \cdot \vec{x_j} + C) \text{ for some } \gamma > 0 \\ c < 0\end{gathered} \tag{7}$$

When utilizing kernels, an extra hyper-parameter, $\gamma$, is also necessary. This controls how much influence data points have on the decision boundary, i.e. a smaller $\gamma$ will allow the hyperplane to consider further away points. Tuning C and $\gamma$ will inform the best combination for the kernel. `GridSearchCV` from `sklearn` [47] will be used to do this; it performs an exhaustive grid-search of specified hyper-parameters and kernels (Table 7 in appendix), and 5-fold cross validation will be used. This will be optimised once for the recall and F1 metrics, and then compared.

### 3.5 Hybrid CNN-SVM

The hybrid CNN-SVM will work by replacing the sigmoid classification layer of the CNN with a SVM. The aim is that the CNN will extract the salient features of the input image, therefore allowing a powerful SVM classifier. After the CNN weights have been updated for our training datatset, the penultimate layer will be extracted. These features will then be used to train the SVM, and the optimal decision boundary found for the specified scoring metric.

Metrics of the model will then be evaluated using the testing dataset. The images will be fed through the CNN, the layer will be extracted, and these features will be classifier by the SVM. This will then be repeated for the second dataset to test for generalisability.

### 3.6 Metrics

Several metrics will be utilized in order to assess the quality of the models. In this project, positive and negative will signify fire and no fire respectively. Accuracy (Equation 8) is a widely used and recognized metric nonetheless is not always appropriate. Risk aversion is important for fire detection, however accuracy assigns equal weight to each error. Precision (Equation 9), is the positive predictive rate and is good metric in situations where the cost of False Positives is high. Recall (Equation 10) is the percentage of positive results correctly classified, and can be used to minimize False

Negatives. Optimizing for this metric can however encourage excessive False Positives, and so must be used carefully. Due to this, the F1 score (Equation 11) will also be used. This is the harmonic mean of precision and recall, and acts as a trade-off between them.

$$Accuracy = \frac{True\ Positives + True\ Negatives}{All\ Samples} \tag{8}$$

$$Precision = \frac{True\ Positives}{True\ Positives\ +\ False\ Positives} \tag{9}$$

$$Recall = \frac{True\ Positives}{True\ Positives\ +\ False\ Negatives} \tag{10}$$

$$F1\ Score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{11}$$

## 4 Results and Discussion

### 4.1 Evaluation of the effects of training the models using augmented images

The three imported models were trained and validated on dataset 1 with initially only the custom classification layer added as a modification. A proportion of all the pre-trained weights of each model were made available for training and the input size, batch size, and learning rate of the initial models can be seen in Table 2. A low learning rate was used since training a proportion of pre-trained weights can lead to overfitting by updating the weights by increments that are too large [48]. All models were trained until convergence. Figure 8 shows the learning curves for MobileNetV3_small and all models followed a similar learning curve. To evaluate and compare the performance of the initial models for fire classification they were then tested on dataset 1. All models performed very well with MobileNetV3_small, EfficientNetB0, and EfficientNetB4 achieving an F1 score of 97.4%, 98.4%, and 98.4% and a recall of 97.9%, 97.9%, 98.4% respectively. As mentioned previously, the images in dataset 1 are considerably clean (they do not contain any human infrastructure), so to test the effects that image augmentation would have on the performance of the models, all three models were re-trained with the same parameters in Table 2 on dataset 1 after it had been passed through the `ImageDataGenerator`. Training the images on augmented data did show improvement in all three of the models performance (Table 3), with EfficientNetB0 performing the best out of the three models achieving an F1 score of 99.2%, recall of 98.4%, precision of 100%, and an accuracy of 99.2%. For the models trained using image augmentation MobileNetV3_small, EfficientNetB0, and EfficientNetB4 achieved a false negative rate (FNR) of 0.021, 0.016, 0.026 respectively out of 381 test images. Despite EfficientNetB4 having considerably more trainable parameters, it miss-classified as many images (six) as the smallest model tested, MobileNetV3.

It was noted during the testing that the type of images the models were miss-classifying as fire were all of sunsets or where the sun is creating strong, orange light in the image (see Figure 9 for examples). It may have been beneficial to train the models on datasets that contain a variety of these sorts of images to overcome this challenge. However, since one of the key aims of this research is to reduce the amount of images containing fires being miss-classified as not fire, a more important metric to try to reduce, a focus on making the models more risk-adverse was taken. The results of this will be discussed in section 4.3.

### 4.2 Evaluation of models performance on an unseen dataset

The models were tested on dataset 2 to asses performance of the models against unseen, or different data. The models that were trained on augmented data performed better than the initial models and the classification report for these are shown in Table 4. Interestingly EfficientNetB4 performed the best across all metrics on this dataset achieving 90.1%, 85.3%, 97.0%, and 90.2% for F1 score, recall, precision and accuracy respectively. A possible reason for this is it's larger parameter size meaning that it is less prone to overfitting and more able to generalise to previously unseen data.

The confusion matrices for all models can be seen in Figure 7. As expected the models miss-classified more images on dataset 2 however performance was generally good. EfficientNetB4 and MobileNetV3_small performed similarly with a FNR of 0.15 and EfficientNetB4 had a slightly better false positive rate (FPR) of 0.04 compared to 0.06. EfficientNetB0, which performed the best on dataset 1 had a higher FNR than the other two models (0.17 compared to 0.15) and a FPR of 0.05.

One thing to note when comparing the performance of these models is the time it takes to train and test the models. Given the end use case of this model would be for use on low-power or mobile devices, the computational resources required to execute these models will affect it's applicability for real-world use cases. Due to it's substantially larger size than the other two models, EfficientNetB4 took considerably longer to train and to test. It's worth noting that while the performance of this model is better on dataset 2 it may not be practical to deploy this model on mobile devices and as such further investigation should be carried out to assess this.

All values are percentages (%) and models with the best performance for each metric are displayed in bold.

### 4.3 Hybrid CNN+SVM evaluation

The proposed hybrid CNN-SVM model was then developed. Replacing the last layer of the CNN with an SVM has decreased the False Negatives, i.e. undetected fires, for all architectures (examples in Table 6), and a full list of models

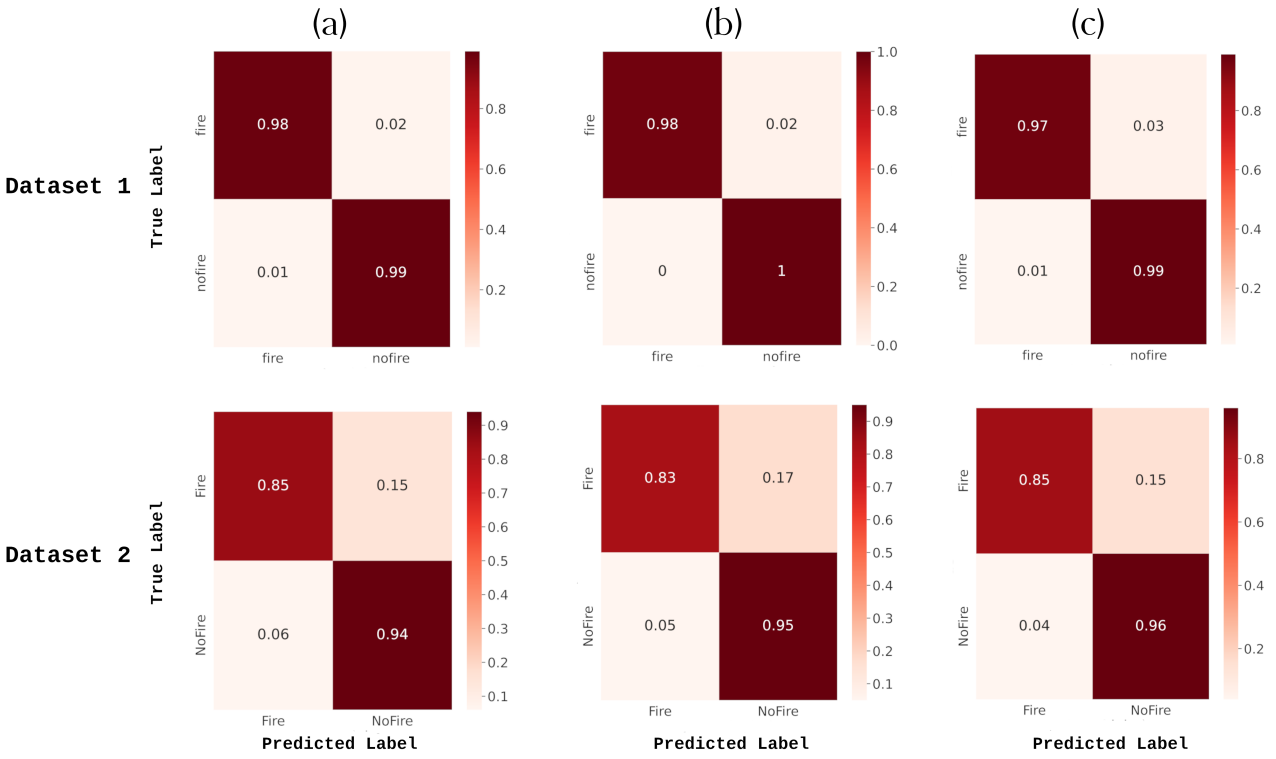


Fig. 7: Confusion matrices for models trained on augmented data for dataset 1 and dataset 2 where (a) is MobileNetV3_small (b) is EfficientNetB0 (c) is EfficientNetB4.

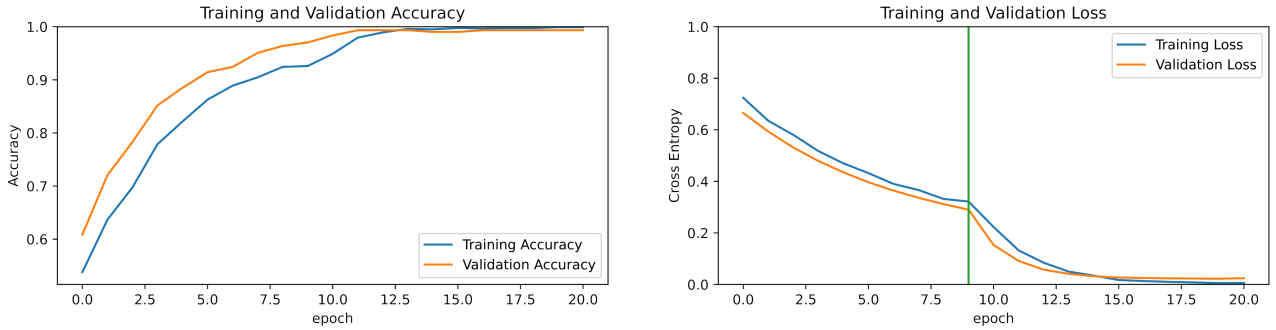


Fig. 8: Accuracy and loss curves for MobileNetV3. The model converges after 13 epochs.

and corresponding metrics can be found in the Appendix (Table 11). An already high recall metric from dataset 1 for the existing architecture is improved further with the use of an SVM at the classification level. The F1 score is also included to provide context, to ensure that the False Positive rate (i.e. false alarms) is not excessive, as this has been an issue with previous fire detection methods. An example of this can be seen when the features extracted from MobileNetV3_small were optimized for recall in the SVM classifier (Table 10 in appendix). It received a perfect False Negative score of 0 when tested on both dataset 1 and 2, however scored poorly in other metrics as the algorithm became too risk-averse and was over-classifying positives.

From all possible models (Table 11 explored in this project, the top 5 were selected for comparison of results (Table 5). These were chosen heuristically for their recall score, whilst retaining high performance in other metrics. Results from both dataset 1 and 2 were considered, and their corresponding hyper-parameters and attributes can be found in the Appendix (Table 9)

Models A and C have perfect recall scores (100 %) in dataset 1, and this remains very high for dataset 2 (97.4% and 99.3 % respectively). However, the F1 scores for dataset 2 are low (89.2 % and 86.6 %) which shows there is a high amount of False Positives, and this is also reflected in their precision (82.2 % and 76.8 %). Model B has a very high recall metric (99.5 %) for dataset 1, and has the highest performance in all other metrics for dataset 1. This not however reflected in the results for dataset 2, suggesting that it may not be very generalizable. Model D performs generally well across all

Table 2: Summary of the training parameters and input conditions for the CNN models

| Architecture | Input size | Batch size | Learning rate | Parameters (million) |
|---|---|---|---|---|
| MobileNetV3_small | 224x224 | 32 | 1e-5 | 0.94 |
| EfficientNetB0 | 224x224 | 32 | 1e-5 | 4.10 |
| EfficientNetB4 | 224x224 | 32 | 1e-5 | 17.68 |

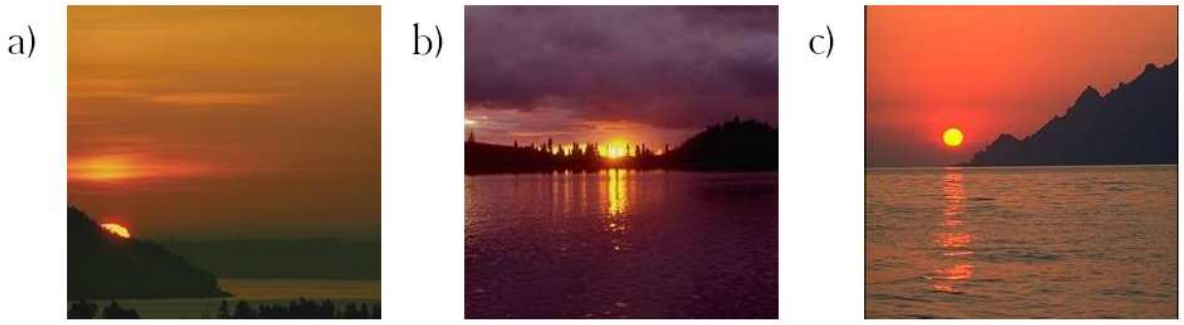


Fig. 9: Example of the images the models miss-classified as fire. All images miss-classified as fire contained the sun, or strong orange light.

Table 3: Performance report for the initial models tested on dataset 1 with and without image augmentation.

| | **Without Image Augmentation** | | | | **Image Augmentation** | | | |
|---|---|---|---|---|---|---|---|---|
| Model | F1 Score | Recall | Precision | Accuracy | F1 Score | Recall | Precision | Accuracy |
| MobileNetV3_Small | 97.4 | 97.9 | 96.8 | 97.4 | 98.4 (+ 1.1) | 97.9 (+ 0) | 98.9 (+ 2.1) | 98.4 (+1.1) |
| EfficientNetB0 | **98.4** | 97.9 | **98.9** | **98.4** | **99.2 (+ 0.8)** | **98.4 (+0.5)** | **100 (+ 1.1)** | **99.2 (+ 0.8)** |
| EfficientNetB4 | **98.4** | **98.4** | 98.4 | **98.4** | 98.4 (+ 0) | 97.4 (- 1.0) | 99.5 (+ 1.1) | 98.4 (+ 0) |

All values are percentages (%) and models with the best performance for each metric are displayed in bold.

metrics for both datasets, retaining over 90% in recall, F1 and accuracy for dataset 2. Although it does not have the best performance, the relatively high metrics on unseen images could still be attractive for low memory devices, as the size is much smaller than that of models that utilize EfficientNetB4 (models C and E). Model E performs the best for precision, F1 and accuracy for dataset 2 (92.8 %, 91.6 % and 91.7 % respectively), however it performs the worst for recall (90.6 %) for these top 5 models. Not detecting 10 % of forest fires could have devastating effects, and so would not be appropriate for this case. The confusion matrices for the top 5 models for dataset 2 can be seen in Figure 10.

This improvement in the recall metric makes a strong case for utilizing this proposed framework, to minimize missed fires and therefore limit damage. Moreover, the models with SVM classifiers outperformed all our selection criteria, further promoting this hybrid approach. The best model for forest fire detection would be dependent on the situation. For example, if the system was deployed as a warning system where a human observer could check camera footage quickly, model C would be the most appropriate, as very little fires would be missed.

## 5 Conclusions and Limitations

The use of CNNs in early forest fire detection systems poses a great opportunity to improve the effectiveness and reliability of the current systems. Furthermore, designing

Table 4: Performance report for the models trained on augmented data tested on dataset 2.

| **Model** | F1 Score % | Recall % | Precision % | Accuracy % |
|---|---|---|---|---|
| MobileNetV3_Small | 89.3 | 84.6 | 94.4 | 88.7 |
| EfficientNetB0 | 89.3 | 83.3 | 96.1 | 88.5 |
| EfficientNetB4 | **90.1** | **85.3** | **97.0** | **90.2** |

Table 5: Metrics for the best 5 models testing on dataset 1 and dataset 2

| Model (reference) | **Dataset 1** | | | | **Dataset 2** | | | |
|---|---|---|---|---|---|---|---|---|
| | Recall | Precision | F1 | Accuracy | Recall | Precision | F1 | Accuracy |
| EfficientNetB0+SVM_R (A) | **100.0** | 97.9 | **99.0** | 98.9 | 97.4 | 82.2 | 89.2 | 88.2 |
| MobileNetV3+SVM_R (B) | 99.5 | **98.4** | **99.0** | **99.0** | 93.0 | 88.2 | 90.5 | 90.3 |
| EfficientNetB4+SVM_R (C) | **100.0** | 92.7 | 96.2 | 96.1 | **99.3** | 76.8 | 86.6 | 84.6 |
| MobileNetV3+SVM_F (D) | 98.4 | 97.4 | 97.9 | 97.9 | 95.4 | 87.1 | 91.1 | 90.7 |
| EfficientNetB4+SVM_F (E) | 98.9 | **98.4** | 98.7 | 98.7 | 90.6 | **92.8** | **91.6** | **91.7** |

All values are percentages (%) and models with the best performance for each metric are displayed in bold. Models are listed in an arbitrary order.

Table 6: Comparison of Recall and F1 scores for different classification methods

| **Classifer >** | CNN | SVM | CNN | SVM |
|---|---|---|---|---|
| **Architecture** | Recall | | F1 | |
| MobileNetV3 | 97.9 | 99.5 (+1.6) | 98.2 | 99.0 (+0.8) |
| EfficientNetB0 | 98.5 | 100.0 (+1.5) | 99.2 | 99.0 (-0.2) |
| EfficientNetB4 | 97.4 | 99.0(+1.6) | 98.4 | 98.7 (+0.3) |

All values are percentages (%), and difference is shown in brackets. Sigmoid classifier is the binary classifier for the CNN. The SVM classifier for MobileNetV3 model uses a linear kernel (C=1) with no augmentation, for EfficientNetB0 uses a linear kernel (C=1) with augmentation, and for EfficientNetB4 uses a RBF kernel (C=1, $\gamma$=0.001) with augmentation. All testing on dataset 1.

LW-CNN models for use in low-energy devices means the benefit of CNNs can be had in areas with limited technological infrastructure. While current literature investigates the effectiveness of LW-CNNs for use in forest-fire detection, updated SOTA models have been released with better performance and efficiency. Therefore, we investigated the performance of these newer models from the MobileNet and EfficientNet architectures and aimed to improve their performance by implementing image augmentation and the use of an SVM classifier to reduce the false negative rate of the models. The models are evaluated on two separate datasets to test their performance on a variety of images.

The imported models performed well with only a custom classification layer added (no image augmentation or SVM) and EfficientNetB4 performed the best with an accuracy and recall of 98.4% when tested on dataset 1. All models saw an improvement from being trained on augmented data with EfficientNetB0 experiencing the largest increase in performance (+ 0.5% recall and +0.8% accuracy). Replacing the custom classification layer with an SVM further increased the models performance. Both EfficientNetB0 and EfficientNetB4 achieved a perfect recall score of 100% (and accuracy of 98.9% and 96.1% respectively) with an SVM classifier optimised for recall however MobileNetV3_small with an SVM classifier optimised for recall had the best accuracy performance (99.0%).

Model performance on dataset 2 dropped, however the models trained on augmented data performed well with an SVM classifier. EfficientNetB4 with an SVM calssifier optimised for F1 had the best accuracy when test on dataset 2 (91.7%) and a recall of 90.6%. EfficientNetB4 with an SVM classifier optimised for recall performed the best on recall (99.3%) however had a relatively low accuracy (84.6%) showing that there is a trade-off between accuracy and recall when trying the make the model more risk-averse.

There did not appear to be a model that performed substantially better than the others and the perceived performance of a model depends on what metrics are being compared. To avoid the potential for fires to go undetected by the models,

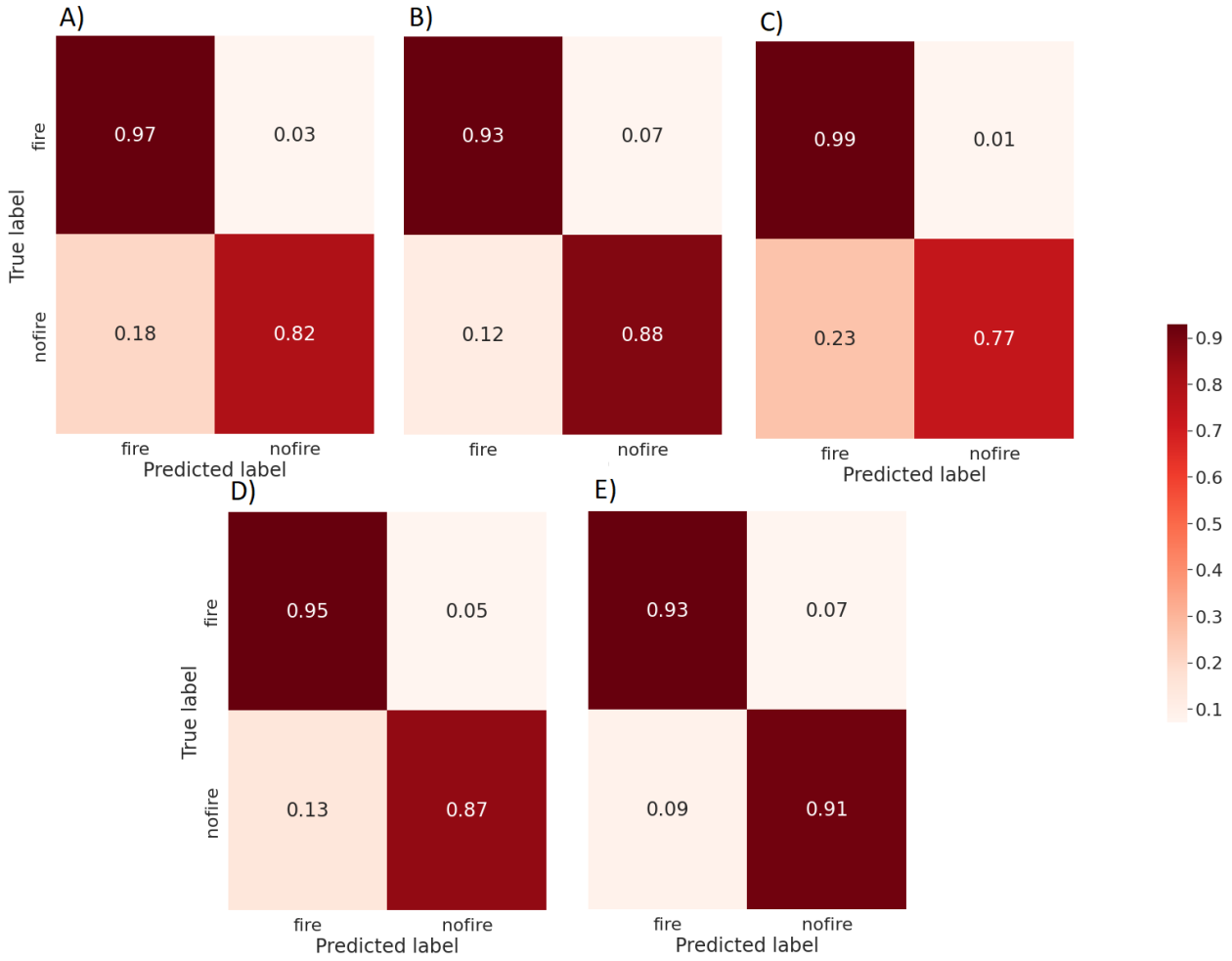


Fig. 10: Confusion matricies for the top 5 models for dataset 2. Where A = EfficientNetB0+SVM_R, B = MobileNetV3+SVM_R, C = EfficientNetB4+SVM_F, D = MobileNetV3+SVM_F, E = EfficentNetB4+SVM_F.

we have considered recall to be an important metric as this reduces the FNR however depending on where the model is deployed, the user may want to optimise for accuracy or precision to reduce the amount of false alarms caused by the detection systems. As such, this analysis shows the potential for using SOTA LW-CNNs for use in fire detection and demonstrates the improvements that can be made with minimal modifications of these models and provides opportunities for these modifications to be made depending on the use case of the model.

One limitation of this research is that the final models were not tested for their computational efficiency or speed. This would be an essential metric to evaluate before the models could be deployed in a fire detection system as it would dictate the hardware and infrastructure that the models could be used on.

In the future we aim to investigate the potential of other classification algorithms such as K-nearest neighbours (KNN). We would address the issue of the models misclassifying images containing a sunset. This could be improved from introducing a different dataset for training that includes these types of images. This would make the model more robust and reduce the number of false alarm incidences. Additionally, smoke is a good indicator of the start of an early forest fire. It would be beneficial for a model to also be able to identify smoke in images as this would allow authorities to be alerted at the earliest signs of a forest fire. This could again be achieved by using a dataset that contains smoke images.

## 6 Contributions

Meg: 1, 2.2, 3.1, 3.4, 3.5, 3.6, 4.3
Charlie: 1, 2.1, 3.2, 3.3, 4.1, 4.2, 5

## 7 Data Availability

Links to the datasets and all related code can be found at `https://github.com/charliegallop/FireIdentificationmCNN`.

Hybrid CNN-SVM:
`https://colab.research.google.com/drive/1EAj17QmZNrTnimDcQYvt6j6ufz9GbIjK?usp=sharing`

https://colab.research.google.com/drive/13x57jYUBWsoo0BBN7i_Q7UCqxB18QVuR?usp=sharing

https://colab.research.google.com/drive/1VuSYzEj2WypZt6aZGiSF0HfIoQHbWpjB?usp=sharing

https://colab.research.google.com/drive/1IpllXJYnXsCkBQL2O0Q1q_Z2Tg5pjFv8?usp=sharing

https://colab.research.google.com/drive/1O3o9LRrVMLp2gAbwxFwh6mKXEQyroaUA?usp=sharing

https://colab.research.google.com/drive/1qfJbsDbq3oX3xCFqa7IiOXJTkYAtb5Rh?usp=sharing

## References


[1] A. M. Gill, "Landscape fires as social disasters: An overview of 'the bushfire problem'," *Global Environmental Change Part B: Environmental Hazards*, vol. 6, no. 2, pp. 65–80, 2005.

[2] Global$_{f}orest_{w}atch$, *Forest fires climate change | effects of deforestation on wildfires | gfw*. [Online]. Available: https://www.globalforestwatch.org/topics/fires/.

[3] *Nasa - wildfires: A symptom of climate change*. [Online]. Available: https://www.nasa.gov/topics/earth/features/wildfires.html.

[4] M. P. NorTh, S. L. STePhens, B. M. Collins, *et al.*, "Reform forest fire management," *Science*, vol. 349, no. 6254, pp. 1280–1281, 2015.

[5] X.-Y. Li, H.-J. Jin, H.-W. Wang, *et al.*, "Influences of forest fires on the permafrost environment: A review," *Advances in Climate Change Research*, vol. 12, no. 1, pp. 48–65, 2021.

[6] G. R. Van der Werf, D. C. Morton, R. S. DeFries, *et al.*, "Co2 emissions from forest loss," *Nature geoscience*, vol. 2, no. 11, pp. 737–738, 2009.

[7] M. Nawaz and D. Henze, "Premature deaths in brazil associated with long-term exposure to pm2. 5 from amazon fires between 2016 and 2019," *GeoHealth*, vol. 4, no. 8, e2020GH000268, 2020.

[8] *Australia in december 2019*. [Online]. Available: http://www.bom.gov.au/climate/current/month/aus/archive/201912.summary.shtml.

[9] M. Kemter, M. Fischer, L. V. Luna, *et al.*, "Cascading hazards in the aftermath of australia's 2019/2020 black summer wildfires," *Earth's Future*, vol. 9, no. 3, e2020EF001884, 2021.

[10] C. Dickman and T. McDonald, "Some personal reflections on the present and future of australia's fauna in an increasingly fire-prone continent," *Ecological Management & Restoration*, vol. 21, no. 2, pp. 86–96, 2020.

[11] S. M. Davey and A. Sarre, *The 2019/20 black summer bushfires*, 2020.

[12] A. I. Filkov, T. Ngo, S. Matthews, S. Telfer, and T. D. Penman, "Impact of australia's catastrophic 2019/20 bushfire season on communities and environment. retrospective analysis and current trends," *Journal of Safety Science and Resilience*, vol. 1, no. 1, pp. 44–56, 2020.

[13] *With costs approaching $100 billion, the fires are australia's costliest natural disaster*. [Online]. Available: https://theconversation.com/with-costs-approaching-100-billion-the-fires-are-australias-costliest-natural-disaster-129433#.

[14] V. Chowdary, M. K. Gupta, and R. Singh, "A review on forest fire detection techniques: A decadal perspective," *Networks*, vol. 4, p. 12, 2018.

[15] A. A. Alkhatib, "A review on forest fire detection techniques," *International Journal of Distributed Sensor Networks*, vol. 10, no. 3, p. 597 368, 2014.

[16] H. Yar, T. Hussain, Z. A. Khan, D. Koundal, M. Y. Lee, and S. W. Baik, "Vision sensor-based real-time fire detection in resource-constrained iot environments," *Computational intelligence and neuroscience*, vol. 2021, 2021.

[17] W. Benzekri, A. E. Moussati, O. Moussaoui, and M. Berrajaa, "Early forest fire detection system using wireless sensor network and deep learning," *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 5, 2020. DOI: 10.14569/IJACSA.2020.0110564. [Online]. Available: http://dx.doi.org/10.14569/IJACSA.2020.0110564.

[18] A. Bouguettaya, H. Zarzour, A. M. Taberkit, and A. Kechida, "A review on early wildfire detection from unmanned aerial vehicles using deep learning-based computer vision algorithms," *Signal Processing*, vol. 190, p. 108 309, 2022.

[19] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[20] C. Szegedy, W. Liu, Y. Jia, *et al.*, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[22] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[23] A. G. Howard, M. Zhu, B. Chen, *et al.*, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[24] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8697–8710.

[25] K. Muhammad, S. Khan, M. Elhoseny, S. H. Ahmed, and S. W. Baik, "Efficient fire detection for uncertain surveillance environment," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 5, pp. 3113–3122, 2019.

[26] M. Szarvas, A. Yoshizawa, M. Yamamoto, and J. Ogata, "Pedestrian detection with convolutional neural networks," in *IEEE Proceedings. Intelligent Vehicles Symposium, 2005.*, 2005, pp. 224–229. DOI: 10.1109/IVS.2005.1505106.

[27] Q. Guo, F. Wang, J. Lei, D. Tu, and G. Li, "Convolutional feature learning and hybrid cnn-hmm for scene number recognition," *Neurocomputing*, vol. 184, pp. 78–90, 2016.

[28] M. Duan, K. Li, C. Yang, and K. Li, "A hybrid deep learning cnn–elm for age and gender classification," *Neurocomputing*, vol. 275, pp. 448–461, 2018.

[29] S. Ahlawat and A. Choudhary, "Hybrid cnn-svm classifier for handwritten digit recognition," *Procedia Computer Science*, vol. 167, pp. 2554–2560, 2020.

[30] X. Wang, Y. Sheng, H. Deng, and Z. Zhao, "Charcnn-svm for chinese text datasets sentiment classification with data augmentation," *International Journal of Innovative Computing, Information and Control*, vol. 15, no. 1, pp. 227–246, 2019.

[31] H. Masnadi-Shirazi and N. Vasconcelos, "Risk minimization, probability elicitation, and cost-sensitive svms," in *ICML*, 2010.

[32] X.-X. Niu and C. Y. Suen, "A novel hybrid cnn–svm classifier for recognizing handwritten digits," *Pattern Recognition*, vol. 45, no. 4, pp. 1318–1325, 2012.

[33] A. Howard, M. Sandler, G. Chu, *et al.*, "Searching for mobilenetv3," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 1314–1324.

[34] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*, PMLR, 2019, pp. 6105–6114.

[35] A. Khan and B. Hassan, "Dataset for forest fire detection," vol. 1, 2020. DOI: 10.17632/GJMR63RZ2R.1.

[36] M. D. Bloice, C. Stocker, and A. Holzinger, "Augmentor: An image augmentation library for machine learning," *arXiv preprint arXiv:1708.04680*, 2017.

[37] F. Chollet *et al.* "Keras." (2015), [Online]. Available: https://github.com/fchollet/keras.

[38] A. Jadon, M. Omama, A. Varshney, M. S. Ansari, and R. Sharma, "Firenet: A specialized lightweight fire & smoke detection model for real-time iot applications," *arXiv preprint arXiv:1905.11922*, 2019.

[39] M. Hussain, J. J. Bird, and D. R. Faria, "A study on cnn transfer learning for image classification," in *UK Workshop on computational Intelligence*, Springer, 2018, pp. 191–202.

[40] M. Shaha and M. Pawar, "Transfer learning for image classification," in *2018 second international conference on electronics, communication and aerospace technology (ICECA)*, IEEE, 2018, pp. 656–660.

[41] L. Yang, S. Hanneke, and J. Carbonell, "A theory of transfer learning with applications to active learning," *Machine learning*, vol. 90, no. 2, pp. 161–189, 2013.

[42] T. Keras, *Keras documentation: Keras Applications*, en. [Online]. Available: https://keras.io/api/applications/ (visited on 05/02/2022).

[43] Larhmam, *File:svm margins.svg*. [Online]. Available: https://commons.wikimedia.org/wiki/File:SVM_margins.svg.

[44] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[45] S. Rogers and M. Girolami, *A first course in machine learning*. Chapman and Hall/CRC, 2016.

[46] M. Hofmann, "Support vector machines-kernels and the kernel trick," *Notes*, vol. 26, no. 3, pp. 1–16, 2006.

[47] F. Pedregosa, G. Varoquaux, A. Gramfort, *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[48] K. Team, *Keras documentation: Transfer learning & fine-tuning*, en. [Online]. Available: https://keras.io/guides/transfer_learning/ (visited on 04/27/2022).

## Appendix

Table 7: Grid-search parameters for optimial SVM hyper-parameter tuning

| **Hyper-parameter** | **Grid options** |
|---|---|
| Kernel | *linear, rbf, sigmoid* |
| C | 1e-07, 1e-06, 1e-05, 1e-04, 1e-03, 1e-02, 1e-01, 1, 10, 100, 1000, 10000 |
| $\gamma$ | 1e-04,1e-03,1e-02,1e-01 |

Table 8: Possible arguments and corresponding ranges for image augmentation

| Argument | Range |
|---|---|
| Flip | Horizontal/Vertical |
| Zoom | 0.2 |
| Width shift | 0.2 |
| Height shift | 0.2 |
| Brightness | [0.75,1.25] |
| Roation | 30° |

Table 9: Grid-search hyper-parameter results, corresponding scoring metric, and augmentation status for top 5 models

| Architecture | **Hyper-parameters** | | | Scoring metric | Augmented | Model name |
|---|---|---|---|---|---|---|
| | C | kernel | $\gamma$ | | | |
| EfficientNetB0 | 1 | linear | n/a | Recall | Yes | EfficientNetB0+SVM_R |
| MobileNetV3_small | 1 | linear | n/a | Recall | No | MobileNetV3+SVM_R |
| EfficientNetB4 | 1e-7 | sigmoid | 0.01 | Recall | Yes | EfficientNetB4+SVM_F |
| MobileNetV3_small | 1 | RBF | 0.001 | F1 | Yes | MobileNetV3+SVM_F |
| EfficientNetB4 | 1 | RBF | 0.0001 | F1 | Yes | EfficientNetB4+SVM_F |

Table 10: All hybrid CNN-SVM models and metrics

| | | | | | | **Dataset 1** | | | | **Dataset 2** | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Architecture | Augmented | Optimiser | C | Kernel | Gamma | Recall | Precision | F1 | Accuracy | Recall | Precision | F1 | Accuracy |
| **Mobile** | **No** | **Recall** | **1** | **Linear** | | **0.995** | **0.984** | **0.990** | **0.989** | **0.930** | **0.882** | **0.905** | **0.903** |
| B0 | No | Recall | 0.1 | RBF | 0.01 | 0.995 | 0.979 | 0.987 | 0.987 | 0.999 | 0.539 | 0.700 | 0.572 |
| B4 | No | Recall | 10 | Linear | | 0.995 | 0.974 | 0.984 | 0.984 | 0.845 | 0.918 | 0.880 | 0.885 |
| Mobile | No | f1 | 0.001 | Linear | | 0.979 | 0.984 | 0.982 | 0.982 | 0.930 | 0.882 | 0.905 | 0.903 |
| B0 | No | f1 | 10 | Sigmoid | 0.001 | 0.987 | 0.979 | 0.987 | 0.987 | 0.967 | 0.840 | 0.899 | 0.892 |
| B4 | No | f1 | 10 | RBF | 0.001 | 0.989 | 0.974 | 0.982 | 0.982 | 0.889 | 0.911 | 0.900 | 0.901 |
| Mobile | Yes | Recall | 0.0000001 | RBF | 0.01 | 1.000 | 0.742 | 0.852 | 0.826 | 1.000 | 0.509 | 0.675 | 0.518 |
| **B0** | **Yes** | **Recall** | **1** | **Linear** | | **1.000** | **0.979** | **0.990** | **0.989** | **0.974** | **0.822** | **0.892** | **0.882** |
| **B4** | **Yes** | **Recall** | **0.0000001** | **Sigmoid** | **0.01** | **1.000** | **0.927** | **0.962** | **0.961** | **0.993** | **0.768** | **0.866** | **0.846** |
| **Mobile** | **Yes** | **f1** | **1** | **RBF** | **0.001** | **0.984** | **0.974** | **0.979** | **0.979** | **0.954** | **0.871** | **0.911** | **0.907** |
| B0 | Yes | f1 | 0.01 | Linear | | 1.000 | 0.979 | 0.990 | 0.989 | 0.975 | 0.821 | 0.891 | 0.881 |
| **B4** | **Yes** | **f1** | **1** | **RBF** | **0.0001** | **0.989** | **0.984** | **0.987** | **0.987** | **0.906** | **0.928** | **0.916** | **0.917** |

Metrics from all hybrid CNN-SVM models for dataset 1 and 2, including model architecture (Mobile=MobileNetV3_Small,B0=EfiicientNetB0, B4=EfficientNetB4), whether the images have been augmented, the specified optimizer for the grid-search and corresponding output. The chosen top 5 models are shown in bold, and each individual column has been formatted as a heat map for visual aid of metric comparison.

Table 11: Full list of results for all models and datasets

| Model | AI | Classification | DS | Optimizer | C | Kernel | Gamma | Recall | Precision | F1 | Accuracy |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Mobile | No | Sigmoid | 1 | | | | | 0.979 | 0.968 | 0.974 | 0.974 |
| B0 | No | Sigmoid | 1 | | | | | 0.979 | 0.989 | 0.984 | 0.984 |
| B4 | No | Sigmoid | 1 | | | | | 0.984 | 0.984 | 0.984 | 0.984 |
| Mobile | Yes | Sigmoid | 1 | | | | | 0.979 | 0.989 | 0.984 | 0.984 |
| B0 | Yes | Sigmoid | 1 | | | | | 0.984 | 1.000 | 0.992 | 0.992 |
| B4 | Yes | Sigmoid | 1 | | | | | 0.974 | 0.995 | 0.984 | 0.984 |
| Mobile | No | SVM | 1 | Recall | 1 | Linear | | 0.995 | 0.984 | 0.990 | 0.989 |
| B0 | No | SVM | 1 | Recall | 0.1 | RBF | 0.01 | 0.995 | 0.979 | 0.987 | 0.987 |
| B4 | No | SVM | 1 | Recall | 10 | Linear | | 0.995 | 0.974 | 0.984 | 0.984 |
| Mobile | No | SVM | 1 | F1 | 0.001 | Linear | | 0.979 | 0.984 | 0.982 | 0.982 |
| B0 | No | SVM | 1 | F1 | 10 | Sigmoid | 0.001 | 0.987 | 0.979 | 0.987 | 0.987 |
| B4 | No | SVM | 1 | F1 | 10 | RBF | 0.001 | 0.989 | 0.974 | 0.982 | 0.982 |
| Mobile | Yes | SVM | 1 | Recall | 1E-07 | RBF | 0.01 | 1.000 | 0.742 | 0.852 | 0.826 |
| B0 | Yes | SVM | 1 | Recall | 1 | Linear | | 1.000 | 0.979 | 0.990 | 0.989 |
| B4 | Yes | SVM | 1 | Recall | 1E-07 | Sigmoid | 0.01 | 1.000 | 0.927 | 0.962 | 0.961 |
| Mobile | Yes | SVM | 1 | F1 | 1 | RBF | 0.001 | 0.984 | 0.974 | 0.979 | 0.979 |
| B0 | Yes | SVM | 1 | F1 | 0.01 | Linear | | 1.000 | 0.979 | 0.990 | 0.989 |
| B4 | Yes | SVM | 1 | F1 | 1 | RBF | 0.0001 | 0.989 | 0.984 | 0.987 | 0.987 |
| Mobile | No | Sigmoid | 2 | | | | | 0.852 | 0.926 | 0.887 | 0.882 |
| B0 | No | Sigmoid | 2 | | | | | 0.780 | 0.974 | 0.866 | 0.849 |
| B4 | No | Sigmoid | 2 | | | | | 0.876 | 0.944 | 0.909 | 0.905 |
| Mobile | Yes | Sigmoid | 2 | | | | | 0.846 | 0.945 | 0.893 | 0.887 |
| B0 | Yes | Sigmoid | 2 | | | | | 0.834 | 0.962 | 0.893 | 0.885 |
| B4 | Yes | Sigmoid | 2 | | | | | 0.854 | 0.969 | 0.908 | 0.902 |
| Mobile | No | SVM | 2 | Recall | 1 | Linear | | 0.930 | 0.882 | 0.905 | 0.903 |
| B0 | No | SVM | 2 | Recall | 0.1 | RBF | 0.01 | 0.999 | 0.539 | 0.700 | 0.572 |
| B4 | No | SVM | 2 | Recall | 10 | Linear | | 0.845 | 0.918 | 0.880 | 0.885 |
| Mobile | No | SVM | 2 | F1 | 1 | Linear | | 0.930 | 0.882 | 0.905 | 0.903 |
| B0 | No | SVM | 2 | F1 | 10 | Sigmoid | 0.001 | 0.967 | 0.840 | 0.899 | 0.892 |
| B4 | No | SVM | 2 | F1 | 10 | RBF | 0.01 | 0.889 | 0.911 | 0.900 | 0.901 |
| Mobile | Yes | SVM | 2 | Recall | 1E-07 | RBF | 0.01 | 1.000 | 0.509 | 0.675 | 0.518 |
| B0 | Yes | SVM | 2 | Recall | 1 | Linear | | 0.974 | 0.822 | 0.892 | 0.882 |
| B4 | Yes | SVM | 2 | Recall | 1E-07 | Sigmoid | 0.01 | 0.993 | 0.768 | 0.866 | 0.846 |
| Mobile | Yes | SVM | 2 | F1 | 1 | RBF | 0.001 | 0.954 | 0.871 | 0.911 | 0.907 |
| B0 | Yes | SVM | 2 | F1 | 0.01 | Linear | | 0.975 | 0.821 | 0.891 | 0.881 |
| B4 | Yes | SVM | 2 | F1 | 1 | RBF | 0.0001 | 0.906 | 0.928 | 0.916 | 0.917 |

AI = Augmented Images, DS=Dataset, Mobile= MobileNetV3_small, B0=EfficientNetB0, B4=EfficientNetB4