*fire*

MDPI

*Article*

# An Efficient Wildfire Detection System for AI-Embedded Applications Using Satellite Imagery

**George L. James [1,†], Ryeim B. Ansaf [1,2,†], Sanaa S. Al Samahi [3,4], Rebecca D. Parker [1,5], Joshua M. Cutler [1], Rhode V. Gachette [1] and Bahaa I. Ansaf [1,*]**

1 School of Engineering, Colorado State University Pueblo, Pueblo, CO 81001, USA; gl.james1@pack.csupueblo.edu (G.L.J.); r.ansaf@csupueblo.edu (R.B.A.)
2 Department of Biology, Colorado State University Pueblo, Pueblo, CO 81001, USA
3 Al-Khwarizmi College of Engineering, University of Baghdad, Baghdad 10071, Iraq
4 EECS Department, University of Missouri, Columbia, MO 65201, USA
5 Industrial Engineering Technology Department, University of Southern Mississippi, Hattiesburg, MS 39406, USA
* Correspondence: bahaa.ansaf@csupueblo.edu
† These authors contributed equally to this work.

**Abstract:** Wildfire risk has globally increased during the past few years due to several factors. An efficient and fast response to wildfires is extremely important to reduce the damaging effect on humans and wildlife. This work introduces a methodology for designing an efficient machine learning system to detect wildfires using satellite imagery. A convolutional neural network (CNN) model is optimized to reduce the required computational resources. Due to the limitations of images containing fire and seasonal variations, an image augmentation process is used to develop adequate training samples for the change in the forest's visual features and the seasonal wind direction at the study area during the fire season. The selected CNN model (MobileNet) was trained to identify key features of various satellite images that contained fire or without fire. Then, the trained system is used to classify new satellite imagery and sort them into fire or no fire classes. A cloud-based development studio from Edge Impulse Inc. is used to create a NN model based on the transferred learning algorithm. The effects of four hyperparameters are assessed: input image resolution, depth multiplier, number of neurons in the dense layer, and dropout rate. The computational cost is evaluated based on the simulation of deploying the neural network model on an Arduino Nano 33 BLE device, including Flash usage, peak random access memory (RAM) usage, and network inference time. Results supported that the dropout rate only affects network prediction performance; however, the number of neurons in the dense layer had limited effects on performance and computational cost. Additionally, hyperparameters such as image size and network depth significantly impact the network model performance and the computational cost. According to the developed benchmark network analysis, the network model MobileNetV2, with 160 × 160 pixels image size and 50% depth reduction, shows a good classification accuracy and is about 70% computationally lighter than a full-depth network. Therefore, the proposed methodology can effectively design an ML application that instantly and efficiently analyses imagery from a spacecraft/weather balloon for the detection of wildfires without the need of an earth control centre.

**Keywords:** wildfire; remote sensing; machine learning; satellite imagery; convolutional neural network



**Citation:** James, G.L.; Ansaf, R.B.; Al Samahi, S.S.; Parker, R.D.; Cutler, J.M.; Gachette, R.V.; Ansaf, B.I. An Efficient Wildfire Detection System for AI-Embedded Applications Using Satellite Imagery. *Fire* **2023**, *6*, 169. https://doi.org/10.3390/fire6040169

Academic Editors: Aqil Tariq and Na Zhao

Received: 20 March 2023
Revised: 17 April 2023
Accepted: 17 April 2023
Published: 20 April 2023



## 1. Introduction

### 1.1. Wildfire Occurrence and Significant of Early Detection

According to the National Interagency Fire Center [1], there have been over 60,000 wildfire occurrences in the US in 2021 alone. These fires burned over 8.3 million acres of land and caused significant damage to both natural and human-made structures. The primary causes

of these wildfires were human activity, such as unattended campfires, improper discard of cigarettes, and natural causes such as lightning and drought. The devastating impact of these fires highlights the importance of proper wildfire prevention and management efforts.

In addition, the Colorado Division of Homeland Security and Emergency Management [2] also emphasized that Colorado has experienced a substantial increase in wildfires over the past decade, with an annual average of more than 2000 wildfires. The state has seen particularly destructive wildfires recently, including the 2018 Spring Creek Fire, which burned over 108,000 acres, and the 2020 Cameron Peak Fire, which burned over 208,000 acres. The combination of high winds, dry conditions, and increased development in wildfire-prone areas have contributed to the increased wildfire risk in Colorado.

Response time is also a crucial element in controlling wildfire spread and damage. Wildfires are generally estimated to increase at an average of 14.27 miles every hour the fire team delays their response. Previously, California wildfires were reported to grow 400% in coverage area in just 4 h. An example of wildfire speed importance is the 2017 California Thomas Fire which spread so fast that it travelled at a rate equivalent to one football field per second [3]. All these factors contribute to the interest in the study application.

Areas considered hot spots for monitoring and inspecting wildfires have become hard to predict, especially after natural disasters that have become the norm during the past few years [3,4]. As a response, various safety measures, protocols, and technologies have been developed and adopted by authorities and agencies. However, achieving the safety goal depends on how fast the authorities detect and assess the wildfire situation. In addition, the critical zones affected have changed dramatically due to climate change. Currently, most wildfires are found when they become large enough to gain the attention of locals, but the sooner these instances are discovered, the easier they are to extinguish. In 2021, nearly a thousand wildfires were reported in the United States, affecting over 7 million acres of terrain [3,5]. Wildfires can travel up to 14 mph, which reinforces the desire for speed in the response time.

Implementing inspection along the thousands of miles of vegetarian is both costly and timely. Faster detection and evaluation of wildfire incident type and size can allow authorities to allocate the necessary resources to resolve the situation quickly. Several technologies, such as sensor networks and aerial and space imagery for critical regions, are available for remote inspection and monitoring. The inspection process must be implemented and evaluated at pre-selected critical areas or incorporated as part of a routine inspection system at identified hot spots. Thus, any abnormal incidents need more time to be discovered and assessed. In addition, deploying a remote sensing system is expensive and requires significant technical efforts.

A comprehensive, environment-friendly, autonomous system that identifies possible wildfires would be necessary for local communities to be safe. In addition, an early alarm system will also detect and locate potential wildfires that need more attention/inspection. To increase safety and minimize the destruction caused by wildfires, we propose an efficient machine learning model for wildfire detection.

### *1.2. Wildfire Detection and Monitoring Using Machine Learning*

In recent years, satellite imagery and machine learning (ML) has been well adopted to monitor infrastructure changes, natural disasters outcomes, and variations in vegetarian areas [6–11].

One approach that has been proposed is the use of deep learning to train convolutional neural networks (CNNs) to classify satellite imagery. Deep learning techniques have been widely used in image classification and detection and have shown promising results in various applications. In forest fire classification and detection, deep learning can effectively analyse satellite images and identify areas likely to be affected by fires.

The first step in this process is to gather a large dataset of satellite images, which can be collected from various sources such as NASA's Earth Observing System or other commercial satellite providers [12]. These images should include both forest fires and areas

unaffected by fires. Next, the images should be pre-processed and labelled according to whether they contain a forest fire. This can be performed manually by experts or with the help of machine learning algorithms. Once the dataset is prepared, it can be utilized to train a deep learning model using techniques such as convolutional neural networks (CNNs) or recurrent neural networks (RNNs). The model can then be fine-tuned and tested on a separate validation dataset to evaluate its performance. Finally, the resulting NN model can classify and detect forest fires in real-time satellite images to predict the likelihood of a fire occurring in the study area.

Overall, deep learning techniques have the potential to significantly improve the accuracy and efficiency of forest fire classification and detection in satellite images, helping to prevent and mitigate the impacts of these disasters [13–27]. In addition, by automating the process of detecting fires, these approaches can help reduce the impact of wildfires on communities and ecosystems.

The proposed solution [13] shows that combining the Inception-v3 convolutional neural network and the local binary pattern algorithm can provide a reliable and accurate system for the classification and detection of fires in satellite images.

Improving NN performance and efficiency has driven several research works and projects. However, improving NN performance is often costly, especially when deploying the NN model on mobile or stand-alone AI-embedded applications. Moreover, modern state-of-the-art networks require high-computational resources beyond the capabilities of many mobiles and embedded applications. One of the most efficient ML tools for mobile applications is MobileNetV2 [28]. MobileNetV2 is specifically tailored for mobile and resource-constrained environments by significantly decreasing the number of operations and memory needed while retaining the same accuracy. Generally, using MobileNetV2 for wildfire detection serves as a promising method for detecting and managing wildfires in natural habitats efficiently and accurately. Wu H et al. [27] used the MobileNetV2 architecture to develop a CNN for wildfire detection. A data augmentation technique, such as adding Gaussian blur or additive Gaussian noise, was used to simulate extreme scenarios and train the modified MobileNetV2 model. The pre-trained MobileNetV2 architecture and the effective data augmentation pipeline implementation allow effective transfer learning to specified classification tasks while reducing computational complexity and maintaining high accuracy.

Designing and implementing an efficient neural network model for embedded applications is a comprehensive contemporary research field. In addition, developing an ML system that can process spacecraft images locally will reduce the time and cost for transferring information from spacecraft. This includes reducing the computational cost required to encode and decode the spacecraft's information then transmission to the deep space network (DSN) [29]. Thus, sending image data needs high energy/time for encoding, transmitting, and then boosting and decoding the weak signal at DNS centres compared with sending only classification result "Fire" or "NoFire". Therefore, this article focuses on optimizing network hyperparameters in wildfire detection despite data limitations (wildfire-containing images).

This article describes a methodology for selecting and optimizing a neural network model that balances the wildfire prediction performance based on satellite imagery using minimum computational resources and limited training data of wildfire images. The developed NN model can be deployed on any system with low-computational resources/power, such as a small remote sensing satellite or a high-altitude weather balloon. In addition, the optimized neural network model can also be deployed on smartphones as an active real-time, low-cost wildfire-monitoring application.

The proposed work aims to develop an efficient ML system that can classify the forest satellite imagery as (Fire) and (No Fire) for wildfire alarm systems. The alarm system can be deployed through mobile or other embedded applications. First, the ML model is trained to learn the essential features of the forest area based on satellite images. The training phase is performed to be as effective as possible regarding classification accuracy. At the

same time, the final neural network model needs to be light and fast in terms of storage and computational requirements to be suitable for mobile or other embedded applications. Then the trained ML model can be used to identify changes in the area, such as smoke, to auto-detect wildfires. Ultimately, this will decrease the damage that could occur faster than alternative means of detection.

## 2. Data and Methods

### *2.1. Dataset*

In this work, the desired dataset for transfer learning to detect wildfire is achieved by the following steps:

- Step 1: Manage access to near real-time orbital satellite imagery from a global provider: Currently, several commercial satellites scan most places on earth several times (up to 10 times) daily with a resolution of 30–50 cm [12]. Thus, a well-trained ML system can detect objects or changes with a considerable size (mass) that can be considered a wildfire. This task includes investigating the available sources of images, resolution, availability, and cost for various satellite imagery sources. The selected data source for this work is the Sentinel 2 satellite, an European Space Agency (ESA) satellite, from Soar Earth Ltd. [12]. The Sentinel 2 satellite revisits the same location in approximately five days, which is unsuitable for wildfire detection applications. However, for the purpose of developing the method, imagery from this satellite was used. The revisit frequency of the satellite to a particular location depends on the number of satellites in the mission and the orbit altitude from Earth. Therefore, the suggested approach in this article will be more effective when it relays satellite imagery that surveys the same area more frequently and has a revisit frequency of less than one day, such as every 12–24 h or less (WORLDVIEW-3 (<1 day), SKYSAT (3 to 12 times per day with the entire constellation (latitude-dependent) Suomi-NPP (101.44 min) , n.d NOAA VIIRS, Terra (99 min) and Aqua/MODIS, etc.) [30,31]. The availability of the satellite images for locations studied depends on the periodic availability for these various satellite missions. In this work, we focused more on proving the concept of using satellite imagery to predict wildfires based on limited data and computational resources rather than which satellite provides the most daily images of the area.
- Step 2: Determining the study site: The preferred study site is a forest with different landscapes to aide in generalizing the ML model to various land topography. San Isabel National Park, Colorado, US, was selected as the study area. The chosen site has several distributed green zones, mountains, snow, and infrastructures (roads). A sample of the selected images is shown in Figure 1. In addition, the San Isabel National Park area in Colorado is a hotspot for monitoring and inspecting wildfire-related threats early due to climate change and abundance in large-scale wildfires globally.
- Step 3: Collecting and creating training data: Several images should be collected during the daytime in different seasons to consider ground and light variations. Due to the limited availability of satellite images, especially with wildfire, an augmented imaging process has been developed to generate adequate pictures for training the NN model. To generate the datasets to train the CNN, we started with satellite images from the San Isabel National Park for a typical fire season (May–November); then, we developed an algorithm to superimpose the original satellite images with artificial fire images at random locations. Finally, the direction of the smoke was estimated from seasonal wind direction data for the San Isabel forest area during the fire season, changed from north to east [4], as shown in Figure 2.

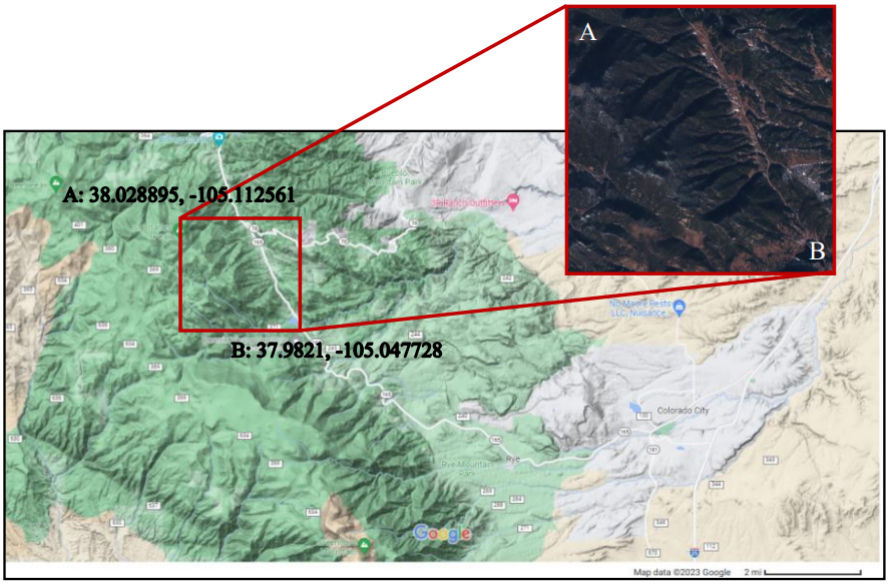


**Figure 1.** Study area latitude and longitude GPS coordinates in decimal degrees (DD).

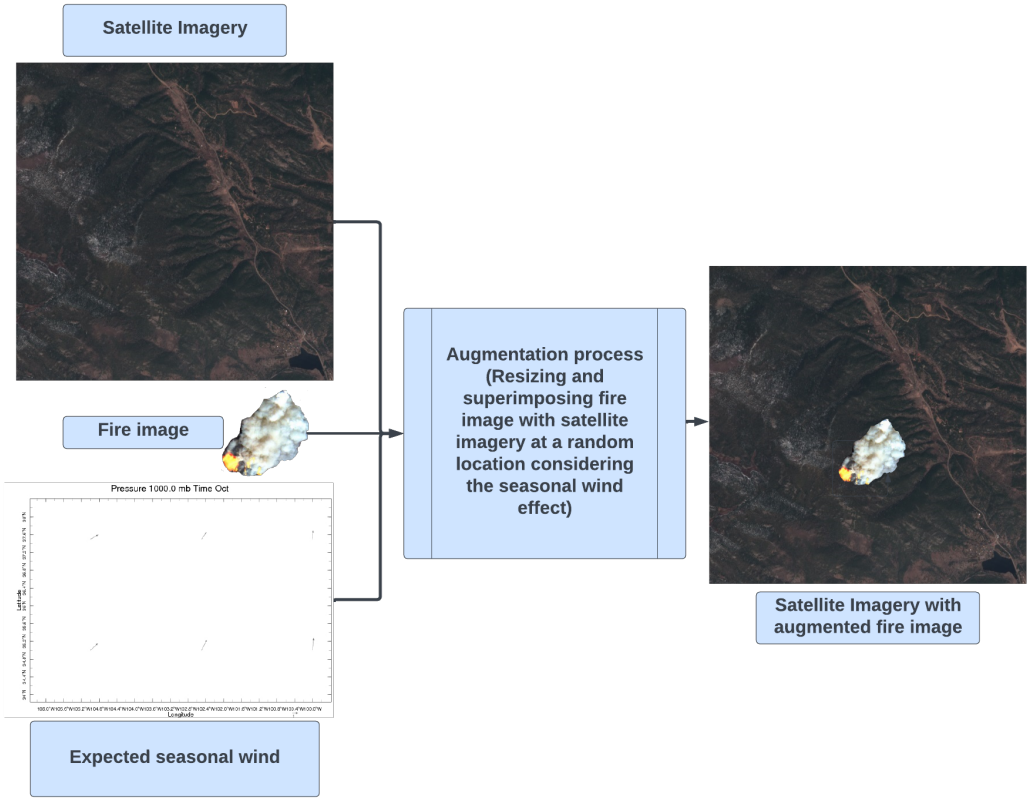


**Figure 2.** Augmentation of fire image in satellite imagery.

### *2.2. Architecture of Object Detection Models and Transfer Learning*

Artificial intelligent (AI) is one of the most pressing topics in contemporary society. Convolutional neural network (CNN) is inspired by the neural network nature of the human brain and can be deployed to create an image classification program. For example, computer vision is also a field in AI that focuses on problems relating to visual cues and images. Combining the CNN into computer vision can serve complex operations ranging from classifying images to solving scientific problems of medicine, astronomy, and even building self-driving cars [6,32,33].

In the developed model, an image is passed through an algorithm, and with every pass, the system learns more features so that future testing is more reliable and accurate. After enough training, the model was able to take a satellite image and give an output of "Fire" or "No Fire", as shown in Figure 3.

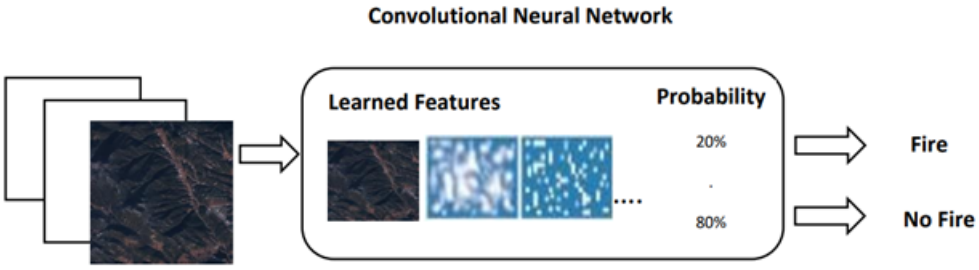


**Figure 3.** Deep learning workflow: Images are passed to the CNN to learn features and classify objects automatically.

A cloud-based platform from Edge Impulse Inc. is used to create a CNN model based on the transferred learning algorithm for the primary task of binary and multi-class, multi-label classification. Recently, pre-trained models such as Inception, VGG16, VGG19, MobileNet, etc., have been created by researchers after training them on millions of images for the purpose of category-based classification [6,12,33–36]. The application of one of these pre-trained images classification models can serve as a base layer. The number of dense layers, output classes, level of accuracy, and computational efficiency are crucial factors that should be addressed carefully to design a successful CNN classifier system.

Additionally, trade-off hyperparameters tailored our system architecture to different performance points. For example, we can adjust the tunable hyperparameters, such as the input image resolution, width multiplier, and drop-out ratio, to obtain the desired performance by the model.

MobileNetV2 is a CNN architecture introduced by Google in 2018 [28]. It was designed for efficient image classification and object detection on mobile devices. The architecture is based on the inverted residuals and linear bottlenecks (IRLB) module, allowing the network to have smaller parameters while maintaining high accuracy. MobileNetV2 also introduces a linear bottleneck layer to reduce computation complexity and improve performance, leading it to be a highly efficient and effective architecture for mobile applications.

This work adopts the transfer learning approach to accelerate the convergence rate and reduce the need for extensive training examples to optimize the weight matrices.

First, we import the MobileNetV2 architecture and its corresponding weights to begin the transfer learning process. Second, we release the final fully connected layer of the model and replace it with a new fully connected layer tailored to our specific classification task. Next, we load our smaller application-specific dataset. Then, we use this dataset to train the modified neural network model. Finally, we evaluate the model's performance.

To achieve high-performance neural network model architecture and generalization, we need to select several hyperparameters for tuning the system to achieve the best performance in terms of classification accuracy and network lightness, such as dropout rate, number of neurons in the final dense layer, and the depth multiplier.

### *2.3. Network Depth Multiplier*

The depth multiplier parameter in MobileNet refers to the scaling factor applied to each convolutional layer's depth (number of channels). Modelling with various channels and computational cost allows for fine-tuning the model's capacity and efficiency. On the other hand, increasing the multiplier is a trade-off between the accuracy, computation and the risk of overfitting in non-linear and complex problems. Using the multiplier parameter in MobileNet allows for flexibility in balancing model performance and efficiency for a given task [28,36].

### *2.4. Dropout Rate*

Dropout is a regularization technique commonly used in deep learning to prevent overfitting. It works by randomly setting a fraction of the input units to zero during training, helping to eliminate the complexity of the model and improve generalization to new data [37]. The effect of dropout on the performance of MobileNet will depend on the specific application and the dropout level used. In general, using dropout can enhance the performance of MobileNet by reducing overfitting and improving the ability of the model to generalize to new data [37]. However, using too high of a dropout level can also negatively impact the model's performance by reducing the network's capacity and preventing it from learning valuable patterns in the data [38].

### *2.5. Dense Layer*

The effect of adding a dense layer to MobileNet will depend on the specific application and the design of the dense layer. A dense layer, also known as a fully connected layer, is one in which all the neurons in the layer are connected to all the neurons in the preceding and following layers. Dense layers are often used in deep learning to learn complex non-linear relationships in the data. Adding a dense layer to MobileNet can improve the network performance by allowing it to learn more complex patterns in the data [38]. However, adding too many dense layers or making the layers too large can also negatively impact the network performance by increasing the parameters and computation time [39,40].

## 3. Network Computational Performance Evaluation

The impact of several hyperparameters on the network computational and classification performances have been investigated. Network computational performance is assessed based on a simulation of the network performance at a specific processing unit. Arduino Nano 33 BLE Sense was used for our analysis. A tiny development board with a Cortex-M4 microcontroller supported by Edge Impulse Inc. was utilized to estimate the output efficiency, specifically, an estimate of inferencing time, peak RAM usage, and Flash usage. We applied this device for comparison between different network architectures. Three computational performances indicators were developed, as follows:

**Flash usage indicator (*FUI*)**: Flash usage represents the maximum flash memory employed if the trained neural network model is deployed on the target device. (i.e., Arduino Nano 33 BLE for our project.) The *FUI* for the network (*i*) is defined as follows:

$$FUI = \frac{Flash\ usage\ for\ network\ (i)}{Flash\ usage\ for\ a\ reference\ network} \quad (1)$$

**Peak RAM usage indicator (*PRI*)**: Peak RAM usage represents the maximum RAM utilized if the trained neural network model is deployed on the target device. The *PRI* for the network (*i*) is defined as follows:

$$PRI = \frac{Peak\ RAM\ usage\ for\ network\ (i)}{Peak\ RAM\ usage\ for\ a\ reference\ network} \quad (2)$$

**Inferencing time indicator (*ITI*)**: Inferencing time represents the time required to obtain an output from the trained neural network model when deployed on the target device. The *ITI* for the network (*i*) is defined as follows:

$$ITI = \frac{Inference\ time\ for\ network\ (i)}{Inference\ time\ for\ a\ reference\ network} \quad (3)$$

Network classification performance was assessed through the test error rate, representing the percentage of testing samples falsely classified in the model testing session.

The optimization goal is to decrease the test error rate while maintaining a lightweight and accurate neural network model (i.e., minimum computational indicator (*FUI*, *PRI*, and *ITI*). Furthermore, to facilitate benchmark comparisons between network models, an overall performance indicator (*OPI*) for the network (*i*) was introduced as follows:

$$OPI(i) = 100 * \left( \frac{(FUI + PRI + ITI)/3}{e^{Test\ error\ rate}} \right) \quad (4)$$

The best network architecture will have the maximum $OPI$.

## 4. Results and Discussions

A Python code for the random algorithms was used to generate artificial wildfire images for the San Isabel National Park study area with an augmented fire image at different locations and wind directions. A total of 614 images were created; 76% for training, and 24% for testing. The dataset was uploaded and analysed using the data explorer on the Edge Impulse Development Studio, as shown in Figure 4. The data explorer shows a complete view of all data in the project. Figure 4 illustrates that all images are labelled and classified correctly, with the need for a non-linear classification network mode.

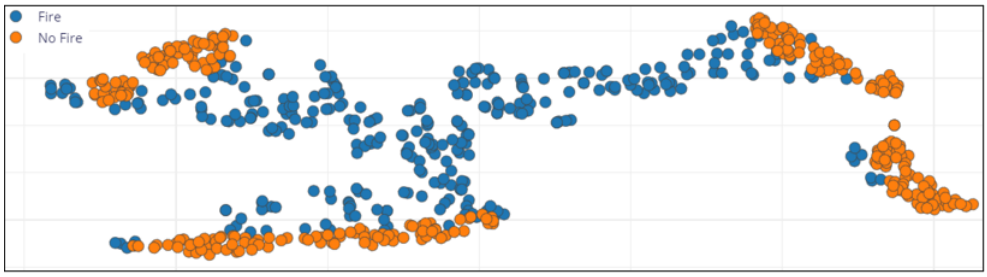


**Figure 4.** Exploring training/testing dataset (64 features).

### *4.1. Effect of Dropout and Number of Neurons in the Dense Layer*

To study the performance effects of the the dropout rate and the number of neurons in a dense layer, on the performance of the neural network model, a MobileNetV2 with a multiplier of 0.35 and image size 96 × 96 was used. The training and testing sessions were implemented for the dropout rate range 0.0–0.2 with an increment step of 0.05 and 0–32 neurons in the dense layer with an increment step of 8. Since the purpose of this study was to compare the effects of hyperparameters, such as dropout and the number of neurons in the dense layer for the same network architecture (i.e., no reference network), the actual values were applied to simulate the network performance on the selected target device (i.e., Flash usage, peak RAM usage, and inferencing time) rather than the performance indicators listed in Equations (1)–(4)).

Figure 5a–d shows the results of this study. The network performance generally increased with the number of neurons in the dense layer. The highest network classification accuracy is achieved when the number of neurons is 32 with a dropout rate of 0.15, as shown in Figure 5a. Furthermore, it was noted that the dropout parameter has a small apparent effect on the network performance with little to no neurons in the dense layer. This type of effect can be related to the application dataset and image resolution, which is relatively low (96 × 96 pixels) and shows a high scattering of two classes (Fire, No Fire), as shown in Figure 4. This makes most of the extracted features at each layer crucial for the classification task so the dropout of some of these features will not guarantee improved network performance. On the other hand, adding more layers will help the network learn complex non-linear relationships in the data.

Figure 5b represents evidence of the direct relationship between Flash usage and the number of neurons on the dense layer with no effect observed when the dropout rate was changed. This was expected since adding more processing units in the dense layer will increase the number of parameters in the network. At the same time, the dropout process was implemented by randomly disabling neurons during the training session only to improve network prediction performance. The disabled neurons will be activated during the testing session since they are still part of the network architecture and requires the same computational space.

Figure 5c shows that the peak RAM usage is increased when a dense layer is added to the network architecture. Again, the dropout had no effect since this parameter was used during the training process and not during on-device neural network model simulation.

Figure 5d highlights that the variation in the network interference time positively correlates with the increasing number of neurons in the dense layer without affecting the dropout rate. It is noteworthy to mention that adding more processing units in the dense layer increases the network's number of parameters and processing time. In addition, changing the dropout had no effect, and the observations are consistent with those in the above study.

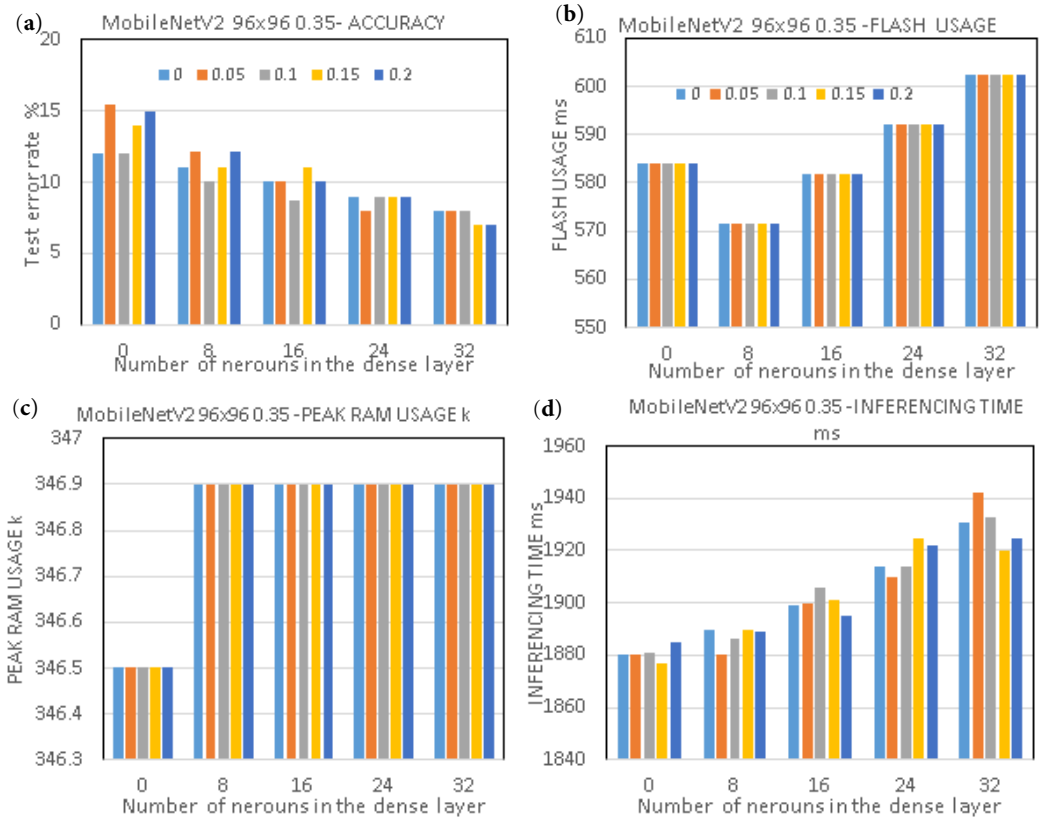


**Figure 5.** Effect of the number of neurons in the dense layer on model accuracy, Flash usage, peak RAM usage, and model inference time.

### *4.2. Effect of the Network Depth Multiplier and Image Resolution*

To study the effect of the network depth multiplier and input image resolution, ten networks were examined and evaluated using the same testing process implemented with the study in Section 4.1. Figure 6 shows that raising the input resolution from 96 × 96 to 166 × 166 pixels highly improved network performance. MobileNetV2 166 × 166 1 has a test error rate of about 2%. In addition, it is noted that reducing the depth multiplier decreases the classification performance of the MobileNet network. Furthermore, it was observed that increasing the dropout rate had no noticeable effect on all MobileNet models tested. Increasing the number of neurons in the dense layer improves network classification accuracy when low input image resolutions (96 × 96 pixels) are used. However, with higher input image resolutions, (166 × 166 pixels), it does not seem to be as affected. Adding a new dense layer improves network performance since the dense layer provides the network with new classification boundaries with a limited number of features extracted from low-resolution images. Increasing the network size and input image resolution improves the classification performance at the cost of increasing the storage environment required for the trained network and makes constraints when the model is deployed on mobile or limited-resource devices. For example, as shown in Figures 7 and 8, all computational indicators *FUI*, *PRI*, and *ITI* were dramatically increased with the use of high-resolution

input images with no depth reduction, e.g., MobileNetV2 16 × 166, chosen as the reference network to calculate all performance indicators, Equations (1)–(3).

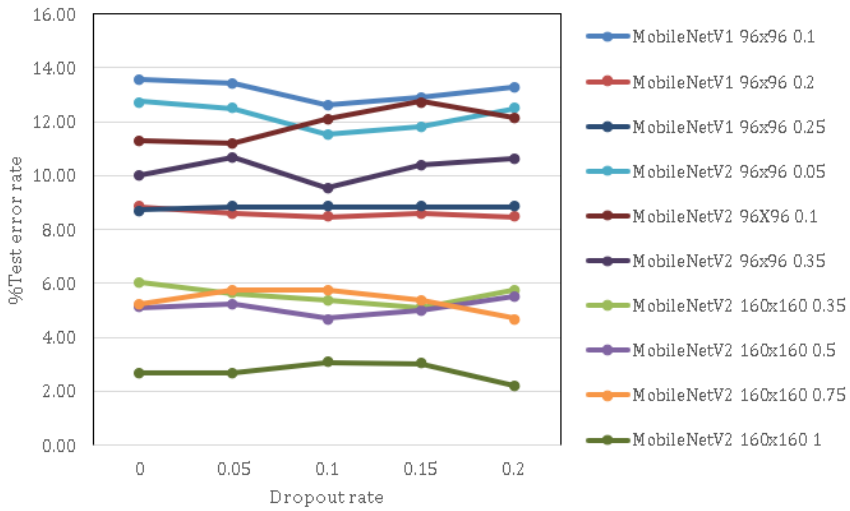


**Figure 6.** Effect of dropout rate on model performance for the tested MobileNet architectures.

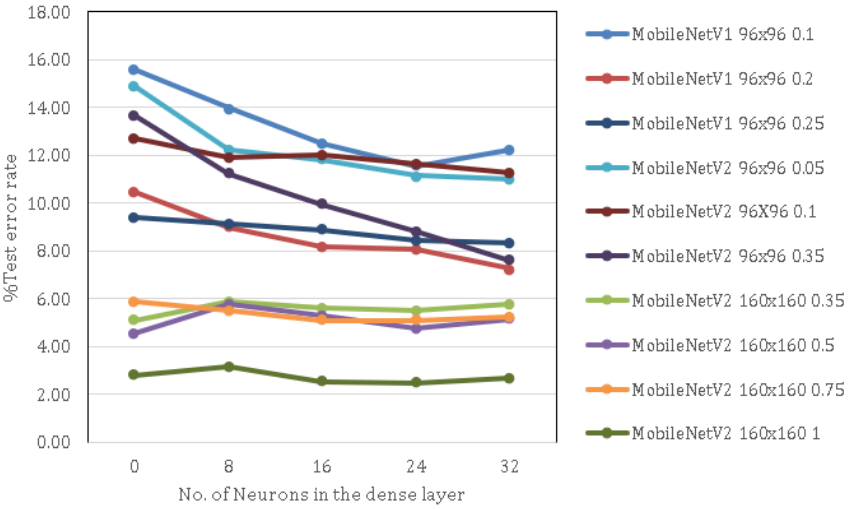


**Figure 7.** Effect of the number of neurons in the dense layer on model performance for the tested MobileNet architectures.

To determine the network accuracy and relative computational cost, the overall performance indicator (Equation (4)) was calculated for all tested networks, as shown in Figure 9. The MobileNetV2 166 × 166 0.5 classified input satellite imagery with an accuracy of 95%. Additionally, it is lighter and faster than MobileNetV2 166 × 166 by about 65%. Therefore, the developed NN model for wildfire alarms can be deployed with reasonable accuracy on limited-capacity mobile devices for the provided wildfire detection application using this version.

In general, increasing the size of a model (e.g., increasing the number of neurons or layers) can lead to improved performance on the training set because the model has more capacity to fit the data. However, this can also increase the risk of overfitting when the model is too complex and begins to fit noise in the data rather than the underlying signal. On the other hand, decreasing the model size can reduce the risk of overfitting butalso decrease the model's ability to capture the relevant patterns in the data, leading to reduced performance. Therefore, it is essential to balance model capacity and overfitting by carefully tuning the multiplier and other hyperparameters as shown in this study.

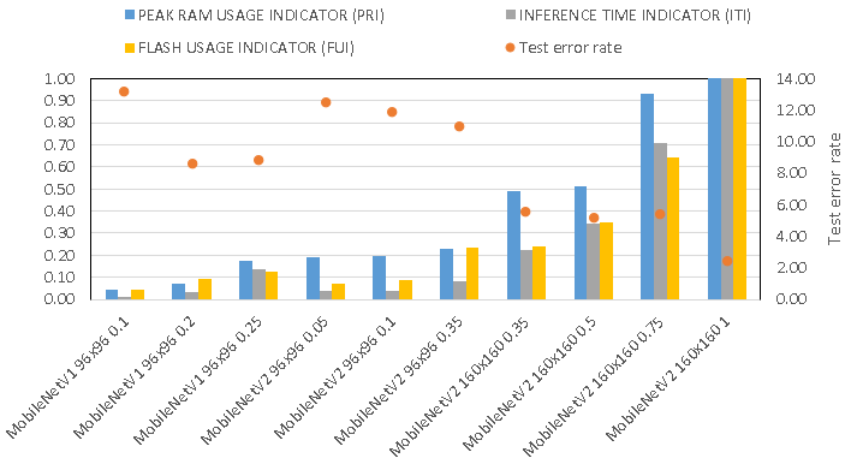


**Figure 8.** Computational indicators for the tested MobileNet architectures.

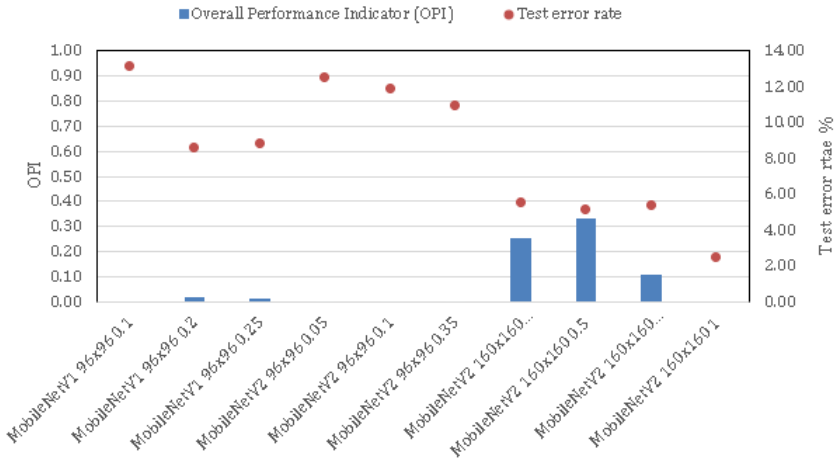


**Figure 9.** Overall performance indicators for the tested MobileNet architectures.

## 5. Conclusions

This paper introduces a systematic approach to selecting a neural network architecture tailored to mobile and resource-constrained environments. The chosen network significantly decreases the number of operations and memory needed while retaining high accuracy, i.e., the required computational cost. In addition, image augmentation was used to overcome the limitation of training examples for wildfire satellite imagery. Several MobileNetV1 and MoileNetV2 networks were tested based on input image resolution and network hyperparameters (dropout rate, network depth, and the number of neurons in the dense layer) to maintain high classification accuracy and on-device computational efficiency. The results show that when the number of neurons in the dense layer increase, the neural network model's computational and storage requirements also increase, therefore resulting in a slower network interference time. At the same time, the network prediction performance did not noticeably improve with these factors. The developed total network performance indicator displays an effective way to select an efficient network architecture and hyperparameters with limited training examples. The network design and hyperparameters were chosen to allow for high prediction accuracy while maintaining an efficient computational process compared with a basic neural network model (MobileNet V2). Thus, the optimized neural network model can be a good option for mobile or embedded AI-applications. The proposed methodology can be used to design a small ML application that can locally and rapidly analyse satellite imagery within the spacecraft and predict the probability of a wildfire existing in the captured images. In turn, this will reduce the size of information that needs to be encoded and transmitted to the DSN centre as

opposed to sending all the image data to Earth to be processed and analysed by the current transmission and communication methods.

**Author Contributions:** Conceptualization, G.L.J., R.D.P., R.B.A. and B.I.A.; methodology, S.S.A.S. and B.I.A.; software, G.L.J., R.D.P. and B.I.A.; validation, G.L.J., S.S.A.S., R.B.A., J.M.C. and B.I.A.; analysis, G.L.J., R.B.A., J.M.C., R.D.P., R.V.G; writing—original draft preparation, S.S.A.S., R.B.A., R.D.P., R.V.G. and B.I.A.; writing—review and editing, S.S.A.S., R.B.A. and B.I.A. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research is supported by The CSU Pueblo Communities to Build Active STEM Engagement (CBASE; /#/ P031C160025) and The Mentoring, Access, and Platforms in STEM (MAPS; /#/ P031C210043); CBASE and MAPS are both Department of Education Hispanic Serving Institution (HSI) STEM Grants (Title III, Part F).

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The authors declare that all data supporting the findings of this study are available in the article and can be provided by the corresponding author upon reasonable request.

**Acknowledgments:** The authors would like to acknowledge the support from Discovery Scholars program at Colorado State University Pueblo. Furthermore, the authors would like to acknowledge the valuable feedback and technical support from Zachary Holden, Cameron Valdez, and Isam Al Atby and the engineering mentoring team (Neb Jaksic, Jude DePalma and Trung Duong) at Colorado State University Pueblo. In addition, we would like to acknowledge the organizations and companies that provided free access to the technical and ML development platform, Edge Impulse Inc., and Google Earth.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. National Interagency Coordination Center Wildland Fire Summary and Statistics Annual Report 2021. Available online: https://www.predictiveservices.nifc.gov/intelligence/intelligence.htm (accessed on 7 January 2023).
2. Home | Division of Homeland Security and Emergency Management. Available online: https://dhsem.colorado.gov (accessed on 6 January 2023).
3. How Wildfires Start and Spread | Mercury Insurance. Available online: https://www.mercuryinsurance.com/resources/weather/how-wildfires-start-and-spread.html (accessed on 7 January 2023).
4. Climatologies. Available online: https://iridl.ldeo.columbia.edu/maproom/Global/Climatologies/index.html (accessed on 9 November 2022).
5. 2021 North American Wildfire Season—Center for Disaster Philanthropy. Available online: https://disasterphilanthropy.org/disasters/2021-north-american-wildfire-season/ (accessed on 9 November 2022).
6. Image Classifier Using CNN—GeeksforGeeks. Available online: https://www.geeksforgeeks.org/image-classifier-using-cnn (accessed on 9 November 2022).
7. Zhang, C.; Pan, X.; Li, H.; Gardiner, A.; Sargent, I.; Hare, J.; Atkinson, P.M. A hybrid MLP-CNN classifier for very fine resolution remotely sensed image classification. *ISPRS J. Photogramm. Remote Sens.* **2018**, *140*, 133–144. [CrossRef]
8. Miranda, E.; Mutiara, A.B.; Ernastuti; Wibowo, W.C. Forest classification method based on convolutional neural networks and sentinel-2 satellite imagery. *Int. J. Fuzzy Log. Intell. Syst.* **2019**, *19*, 272–282. [CrossRef]
9. Satellite-Based Infrastructure Monitoring—LiveEO. Available online: https://live-eo.com (accessed on 5 November 2022).
10. Gillespie, T.W.; Chu, J.; Frankenberg, E.; Thomas, D. Assessment and prediction of natural hazards from satellite imagery. *Prog. Phys. Geogr.* **2007**, *31*, 459–470. [CrossRef] [PubMed]
11. Mccullough, K.; Feng, A.; Chen, M.; Mcalinden, R. Utilizing satellite imagery datasets and machine learning data models to evaluate infrastructure change in undeveloped regions. In Proceedings of the Interservice/Industry Training, Simulation, and Education Conference, ITSEC, Orlando, FL, USA, 30 November–4 December 2020; no. 20269, pp. 1–12.
12. Soar | Discover Your Earth. Available online: https://soar.earth (accessed on 5 November 2022).
13. Priya, R.S.; Vani, K. Deep learning based forest fire classification and detection in satellite images. In Proceedings of the 11th International Conference on Advanced Computing, ICoAC, Chennai, India, 18–20 December 2019; pp. 61–65. [CrossRef]
14. Seydi, S.T.; Saeidi, V.; Kalantar, B.; Ueda, N.; Halin, A.A. Fire-Net: A deep learning framework for active forest fire detection. *J. Sens.* **2022**, *2022*, 8044390. [CrossRef]
15. Copernicus Sentinels Work Together to Monitor Wildfires–Copernicus Sentinels Work Together to Monitor Wildfires— Sentinel Success Stories, June 2018. Available online: https://sentinel.esa.int/web/success-stories/-/copernicus-sentinels-work-together-to-monitor-wildfires (accessed on 6 January 2023).

16. Nolde, M.; Plank, S.; Riedlinger, T. An adaptive and extensible system for satellite-based, large scale burnt area monitoring in near-real time. *Remote Sens.* **2020**, *12*, 2162. [CrossRef]
17. Li, Z.; Cihlar, J.; Nadon, S.; Stocks, B. Satellite-based mapping of Canadian boreal forest fires: Evaluation and comparison of algorithms. *Remote Sens.* **2000**, *21*, 3071–3082. [CrossRef]
18. Govil, K.; Welch, M.L.; Ball, J.T.; Pennypacker, C.R. Preliminary results from a wildfire detection system using deep learning on remote camera images. *Remote Sens.* **2020**, *12*, 166. [CrossRef]
19. Zhang, A.; Zhang, A.S. Real-time wildfire detection and alerting with a novel machine learning approach. *Int. J. Adv. Comput. Sci. Appl. IJACSA* **2022**, *13*. [CrossRef]
20. Fernández, F.G.; Martins, L.; de Almeida, R.V.; Gamboa, H.; Vieira, P. A deep learning based object identification system for forest fire detection. *Fire* **2021**, *4*, 75. [CrossRef]
21. Kasyap, V. L.; Sumathi, D.; Alluri, K.; Reddy CH, P.; Thilakarathne, N.; Shafi, R.M. Early detection of forest fire using mixed learning techniques and UAV. *Comput. Intell. Neurosci.* **2022**, *2022*, 12. [CrossRef] [PubMed]
22. Yuan, C.; Liu, Z.; Zhang, Y. Learning-based smoke detection for unmanned aerial vehicles applied to forest fire surveillance. *J. Intell. Robot. Syst. Theory Appl.* **2019**, *93*, 337–349. [CrossRef]
23. Sudhakar, S.; Vijayakumar, V.; Kumar, C.S.; Priya, V.; Ravi, L.; Subramaniyaswamy, V. Unmanned aerial vehicle (UAV) based forest fire detection and monitoring for reducing false alarms in forest-fires. *Comput. Commun.* **2020**, *149*, 1–16. [CrossRef]
24. Park, M.; Tran, D.Q.; Jung, D.; Park, S. Wildfire-detection method using DenseNet and CycleGAN data augmentation-based remote camera imagery. *Remote Sens.* **2020**, *12*, 3715. [CrossRef]
25. Hong, Z.; et al., Active fire detection using a novel convolutional neural network based on himawari-8 satellite images. *Front. Environ. Sci.* **2022**, *10*, 102. [CrossRef]
26. Ba, R.; Chen, C.; Yuan, J.; Song, W.; Lo, S. SmokeNet satellite smoke scene detection using convolutional neural network with spatial and channel-wise attention. *Remote Sens.* **2019**, *11*, 1702. [CrossRef]
27. Wu, H.; Li, H.; Shamsoshoara, A.; Razi, A.; Afghah, F. Transfer learning for wildfire identification in UAV imagery. In Proceedings of the 54th Annual Conf. on Information Sciences and Systems (CISS), Princeton, NJ, USA, 18–20 March 2020; pp. 1–6. [CrossRef]
28. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.C. MobileNetV2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2020; pp. 4510–4520.
29. Erickson, K. How Does NASA Communicate with Spacecraft? Available online: https://spaceplace.nasa.gov/dsn-antennas/en (accessed on 7 January 2023).
30. Shannell Frazier. Moderate Resolution Imaging Spectroradiomete, MODIS. Available online: https://modis.gsfc.nasa.gov/about/specifications.php (accessed on 9 April 2023).
31. Apollo Mapping | The image hunters, Boulder, Colorado, USA. Available online: https://apollomapping.com/blacksky-satellite-imagery (accessed on 9 April 2023).
32. The MathWorks. What Is a Convolutional Neural Network?—MATLAB & Simulink. The MathWorks, Inc., 2021. Available online: https://www.mathworks.com/discovery/convolutional-neural-network-matlab.html? (accessed on 2 January 2023).
33. Agarwal, Y. Create Your First Image Recognition Classifier Using CNN, Keras and Tensorflow Backend. *Medium.com*, July 2018. Available online: https://medium.com/nybles/create-your-first-image-recognition-classifier-using-cnn-keras-and-tensorflow-backend-6eaab98d14dd (accessed on 6 December 2022).
34. Dropout in Neural Networks—GeeksforGeeks. Available online: https://www.geeksforgeeks.org/dropout-in-neural-networks (accessed on 6 December 2022).
35. MobileNetV1 Explained | Papers with Code. Available online: https://paperswithcode.com/method/mobilenetv1 (accessed on 15 November 2022).
36. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient convolutional neural networks for mobile vision applications. *arXiv* **2017**. arXiv.1704.04861.
37. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *Mach. Learn. Res.* **2014**, *15*, 1929–1958. Available online: http://jmlr.org/papers/v15/srivastava14a.html (accessed on 6 January 2023).
38. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016. Available online: https://books.google.com (accessed on 6 January 2023).
39. Ghadi, N.M.; Salman, N.H. Deep learning-based segmentation and classification techniques for brain tumor MRI: A review. *J. Eng.* **2022**, *28*, 93–112. [CrossRef]
40. Lecun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]