Fire Detection from Surveillance Camera using Neural Networks

Rohan Roy Tharakan (1st Author)
School of Computer Science and Engineering

Associate Professor Chau Lap Pui
Mr Alexander Matyasko
School of Eletrical and Electronic Engineering

Abstract - Every Year about $3 billion is lost annually as property loss due to highway fires. About 500 people died and over 1300 civilians were injured in the United States of America due to highway fires in the year of 2018 alone. Various reports show that fires on roads are a big threat in a lot of different countries. One of the major reasons that fire causes such destruction is the delay in the time to report its occurrence and hence a solution is to detect it immediately and take the required actions as fast as possible. Surveillance cameras are widely installed along roadways, and the numbers are steadily increasing. The aim of this project is to develop an automated algorithm to detect fire from the surveillance camera. Such an automatic algorithm can bring many benefits, for example, it can help respond more effectively to overcrowded regions and vehicle fire on the roads. To implement this algorithm a machine learning approach was used.

A deep neural network was built keeping the accuracy on the training, validation, and test set were used as the metrics. The dataset was created by accumulating images of fire (on-road and elsewhere) and images that did not contain fire but closely resembled it. This was done to help the model could predict more accurately. Moreover, hyperparameter tuning was also performed to generate a model that gives predictions that have very close to human like accuracy. Such an algorithm can be implemented in surveillance cameras in real-time.

Keywords – Fire Detection, Machine Learning, Convolutional Neural Network, Neural Network, Surveillance Camera, Synthetic Data

1 INTRODUCTION

In the era of artificial intelligence and smart devices, designing innovative solutions to enhance safety in public places is a key priority. Fire outbreaks are one of the causes of loss of property and life all around the world. Reporting an occurrence of a fire outbreak is still a manual process, finding a way to automate it can be a potential solution to reducing losses due to fires.

With the increase in the number of surveillance cameras being installed along roadsides, finding a way to use the data from these cameras to report the presence of fire was the objective of this research.

One of the fields of Artificial Intelligence that mainly focuses on making predictions for unseen data based on the patterns observed from the data that has been used to train it is Machine Learning. The machine learning model being trained does not need to be specifically programmed for the unseen data. This serves as a tool to automate this process as it differentiates itself from other methods because it adapts to the various trends and patters in the data.

With the recent developments in technology and specifically machine learning, costs for computation and data storage have become affordable and hence implementing machine learning solutions into daily devices becomes a viable option.

Neural Network (also called an “artificial neural network”) is one set of algorithms used in machine learning for modelling the data using graph of neurons. Neural Networks are based on the working of the human brain, which consists of millions of individual neurons which are connected to each other in a unique fashion. These neurons accept an input and perform some computation on that input and pass on the value to its connected neurons and finally produce an output. Often the networks formed by the connection of these neurons are consists large in number, and hence the models developed on such networks are called deep learning models.

This research paper explores the possibility of detecting fire using the data obtained from surveillance cameras. The dataset consists of equal number of images that contain fire and do not contain fire. The dataset was divided into three smaller subsets, namely the training set, the validation set and the test set. The training set was used to train and develop the deep learning model. The validation set is also used in the training stage to provide a feedback as to when to stop the training of the model. The train set is used to evaluate the performance of the trained model. These subsets are exclusive of each other.

A neural network consists of various hyperparameters which are the high-level properties of a machine learning model. The value of these hyperparameters are altered to improve the overall performance and accuracy of the model.

In this study, the evaluation was based on the accuracy which refers to the fraction of cases that have been correctly classified[1].

This report summarises the observations and discoveries obtained by giving an insight into the details and the thought process involved while formulating a solution.

2 DATASET

The size of the dataset, number of images, used for producing the model is 10,000. Out of total, 5,000 images contained fire while the remaining did not contain fire. The size of an image was 224 pixels x 224 pixels and contained three color channels. Data augmentation techniques like horizontal flipping was applied randomly on the input data. The training, test and validation split was in the ratio 6:2:2.

Out of the 5,000 images that contained fire 2,567 of the images were synthetically generated by placing superimposing images of fire on videos of roads in Singapore, which was meant to simulate data from a surveillance camera. Certain frames from each video was saved as an image. This procedure was carried out in Adobe Photoshop. The remaining data was downloaded from the internet. The subclasses of the images that contained fire are: day (771 images), day-road-synthetic (1,734 images), day-road (526 images), night (940 images), night-road-synthetic (830 images), night-road (198 images). The test data and the validation data for the images that contain fire are composed of images that contain a road.

Out of the 5,000 images that did not contain fire, the images belonged to the following subclasses: buildings (452 images), car crash (252 images), cats (621 images), clouds (420 images), day (133 images), day-road (556 images), dogs (371 images), night (497 images), night-road (684 images), sunset (1,200 images). These images were chosen from the following classes so that the model picks up the exact features of fire and not just the color or the surroundings.

3 METHODOLOGY

Convolutional Neural Networks (CNN) is a class of deep neural networks that are most commonly applied to analyse visual imagery. CNNs employ a mathematical operation called Convolution which is a specialised kind of linear operation [2]. A convolution operation with multiplication is performed on the image. This operation results in reducing the overall size of the image while keeping the important features of the image. However, while using convolution layers in this model, padding was added so as to not reduce the image size and not lose any data on the corners of the image.

Pooling layers are layers that summarise the features in a region of the output generated by the convolution layer. Hence, further operations are performed on summarised output instead of precisely positioned features as generated by the convolution layer. For the implementation of pooling layers in this project, the max pooling was used.

Fully connected layers refer to the layers where the all the neurons in a given layer are connected to all the neurons from the previous layer.

Dropout layer is a layer which is mainly used to prevent overfitting. Dropout is a technique that works by randomly setting the outgoing edges of certain neurons from a layers to 0 at every update of the training phase [3]. The Dropout rate was kept at 0.01 for this project, meaning that 1% of the connections from the previous layer was dropped.

Regularisation is a process of introducing additional information in order to prevent overfitting [4]. For the development of the model for this project, L2 regularisation was used.

Callback functions are functions that can be called when an epoch starts or ends or if a single batch of images are completed processing or new batch of images start processing. For the development of the model for this project, three callback functions from the TensorFlow backend were used, namely EarlyStopping, ModelCheckpoint and ReduceLROnPlateau.

4 EXPERIMENTATION

From preprocessing the input images to evaluating the model, all the steps were performed on a TensorFlow Backend.

Data Input

As mentioned earlier, the dataset, in this study has two major classes, namely Fire and Non-Fire. The images that contained fire were given the label 0 (positive class) and the ones that did not contain fire were labeled as 1 (negative class).

Before starting the evaluation, an important step was to set a threshold value for labelling the predicted classes. This value was chosen to be 0.5. This means that if the predicted probability for an image is less than 0.5 it will be termed to belong to the positive class.

For preprocessing the input data, the pixel values for all color channels of an image were rescaled to by a factor of (1/255). This was done to make the computation easier. Horizontal flipping was randomly performed on the input images to augment the data.

The training, validation and test data was split into mini batches of batch size of 64.

Building the Model

A sequential model with 5 convolutional layers, 4 pooling layers, 4 fully connected layers and 2 dropout layers was used to create the model. The convolution and fully connected layer use the Rectified Linear Units activation functions.

The convolution layers used a (3x3) size filter to perform the convolution operation, while the pooling layers used a (2x2) size filter. The number of different convolution filters increases from 16 to 128, while the overall dimension of the images decreases from (224,224,16) to (12,12,128). This was done similar to the VGG architecture. This is followed by flattening the image and resizing it into a one-dimensional array. This is followed by fully connected layers and dropout layers. The output layer consists of one neuron implementing the sigmoid function. Hence the output consists of a predicted probability, between 0 to 1, of the image containing fire or not.

Binary cross entropy loss function was implemented to compute the loss for the model, and the Adam optimizer was used with 0.001 as the learning rate.

The model was set to run for 100 epochs with 93, 31 steps and 31 steps for the training, validation and test data respectively.

The following callback functions were called after the end of every epoch:

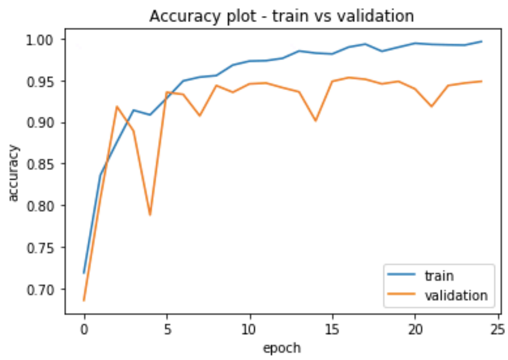
ModelCheckpoint callback function was used to monitor the validation accuracy and to save the model as an H5 file only when the the validation accuracy improved from the previous epoch.

EarlyStopping callback function was used to monitor the validation accuracy and stop the training process when the validation does not improve for eight consecutive epochs. This was implemented to reduce the overfitting on the training data.

ReduceLROnPlateau callback function was used to reduce the learning rate of the model by a factor of 0.1 by monitoring the validation loss. The learning would be reduced if the validation loss does not decrease for five consecutive epochs. This was implemented to allow the model to learn in smaller steps and prevent the increase in the loss.

Running the Model

The model ran for 23 epochs on an average and was stopped by the EarlyStopping callback function. The training accuracy was observed to be increasing throughout the training of the model and the training loss was observed to be strictly decreasing throughout the training phase. However, the validation accuracy and validation loss observed slight deviations from the training accuracy and training loss respectively as shown in the graph



5 RESULTS

After running the model multiple times and tuning the hyperparameters, the average accuracies for the final model are as follows, the training accuracy was found to be 99.8% while the validation accuracy averaged to 95.1% and the test accuracy was evaluated to be 93.5%.

6 CONCLUSION

In conclusion, it is vital to develop measures to prevent the damage caused by fire. Surveillance cameras with the help of computer vision models implementing convolutional neural networks can be used for this purpose. This paper discusses an approach to build to a convolutional neural network to detect fire on roads.

6.1 FUTURE WORKS

The model developed in this study can be further improved and developed to serve more purposes. Due to the reduced data available for training, the model cannot be deployed right away as it lacks robustness. Hence generating more data synthetically and building a proper data input pipeline with more image augmentation techniques would help improve the performance of the model. Determining the accuracy after using transfer learning by using a pretrained model would also be a good strategy to improve the overall robustness of the model.

To put the model to use, predictions on the real-time data coming from a surveillance camera should be sent to the security guard or the person responsible for maintaining the safety of the particular area. Adding a thermal camera to double check the presence of fire would be an efficient way to ensure there are no false warnings.

ACKNOWLEDGMENT

The author would like to express his gratitude to his supervisor Associate Professor Chau Lap Pui, for his constant guidance and support though the project

The author also wishes to extend his gratitude to his mentor Mr Alexander Matyasko from School of Electrical and Electronic Engineering who has been the pillar of support and guidance throughout the extent of this project.

I would like to acknowledge the funding support from Nanyang Technological University – URECA Undergraduate Research Programme for this research project.

REFERENCES

[1] Wikipedia, "Evaluation of binary classifiers", February 2019; https://en.wikipedia.org/wiki/Evaluation_of_binary_classifiers

[2] Wikipedia, "Convolutional neural networks", June 2019; https://en.wikipedia.org/wiki/Evaluation_of_binary_classifiers

[3] Cory Maklin, "Dropout Neural Network Layer In Keras Explained", June 2019; https://towardsdatascience.com/machine-learning-part-20-dropout-keras-layers-explained-8c9f6dc4c9ab

[4] Raimi Karim, "Intuitions on L1 and L2 Regularisation", December 2018; https://towardsdatascience.com/intuitions-on-l1-and-l2-regularisation-235f2db4c261