[Fire No-fire detection]

[LongTimeNoC]

**Data Science Capstone Project-2   
Data Acquisition, EDA and Pre-Processing Report**

Date:

[02/15/2021]

Team Members:

Name: Himanshu Jat

Name: Smith Amornsaensuk

Name: Chingiz Mardanov

Name: Harshit Geddam

## 

### **Introduction**

After all these technological advancements in our society, we still have a dangerous enemy that is one of the most important parts of our life as well- Fire!

To avoid fire breaking out, we have all kinds of hardware equipment in place such as fire alarms, smoke detectors, fire extinguishers etc. still daily in the news we read about the damage caused by fire, whether it be loss of a life or loss of property. As an addition to the current fire prevention system that we have in place, we would like to propose another approach of getting alerts of fire via surveillance cameras. For this capstone project we’re going to design a system which can differentiate between whether there is fire in the image or not. This system will be powered by Deep Learning and can later be installed into surveillance cameras to detect fire.

### **Data Sources**

We have used some of the publicly available datasets in order to train our model. The data that we have obtained contains 1000 images for each class. These are then further split into tets and train sets at a 9:1 ratio respectively. There are 3 different classes in total in the dataset: Fire, Smoke and Neutral. For the purposes of this experiment we are only using two classes: Fire and Neutral.

|  |  |  |
| --- | --- | --- |
|  | Train | Test |
| Fire | 900 | 100 |
| Neutral | 900 | 100 |

Her you can also see a link to the dataset we havel selected: <https://github.com/DeepQuestAI/Fire-Smoke-Dataset>

After we run some of the initial model creation experiments we will then try and extend our dataset with additional images.

### **EDA**

As of the EDA part of the experiment we primarily concentrated on looking at the pixel values for every image. We also analysed their dimensions, orientation and grayscale histogram on our training data.

On the image below you can see the statistics for every individual image and some computations over their pixel values. For example a mean and median pixel value for a certain image. We also show the label of these images and possible range of them.

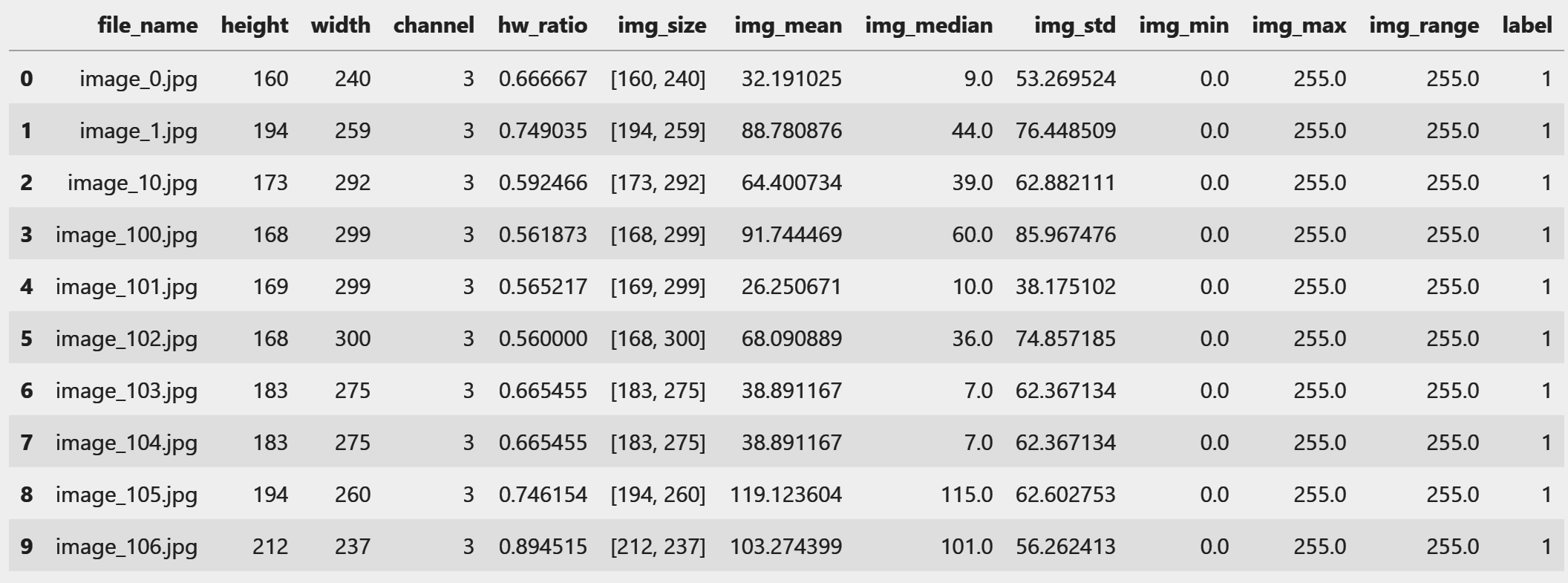


Figure 1: metadata of each image

We also tried to run some analysis on the whole dataset. This is a result of running df.describe on the whole dataset.

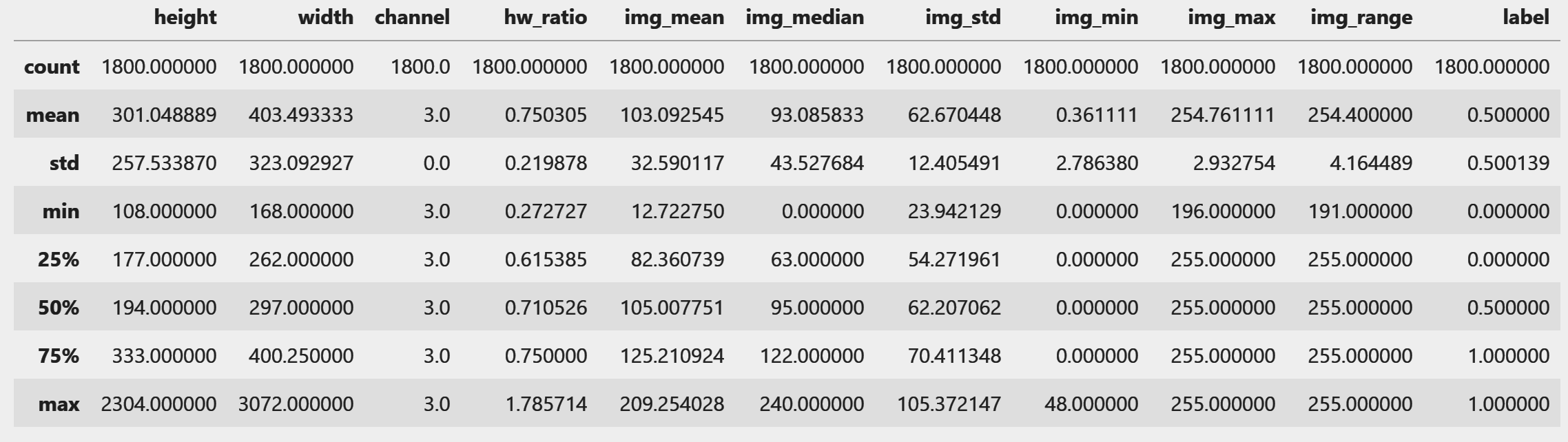


Figure 2: Data description of the whole dataset (1800 images)

We can see that the mean of hw\_ratio (high / width ratio), is 0.75 and std is 0.21. The information tells us that most images are in landscape and they are close to square shape. This brings us to find out more by plotting its histogram.

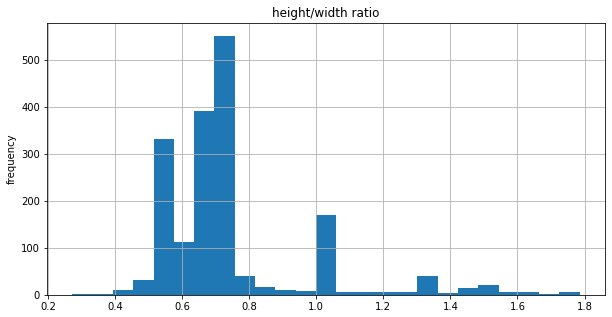


Figure 3: Histogram of height and width ratio

As we expected, most of the images dimension ratio lie between 0.5 and 0.75. We can conclude that most of the images are in landscape and more than half of them have a ratio between 0.65 to 0.75 which are not too far from a square shape. From the information, we consider resizing our image to be square for training purposes.

From Figure 2, we can see that the average height and width are about 300 and 400 in order.

We may choose the rescaling size to be a factor of 300. Looking deeper with histogram will help us to draw the conclusion.

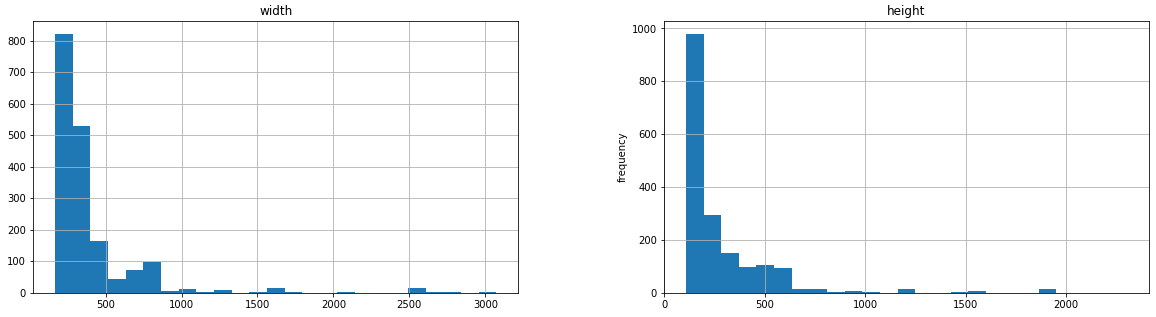


Figure 4: histogram of width (left) and histogram of height (right)

From Figure 4, we can see that most of the images lie around 150 width and height. This information confirms that we can rescale our image to be 150x150.

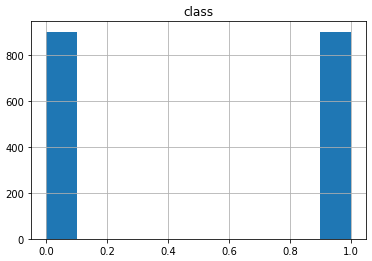


Figure 5: Class distribution

Although the data source has stated the number of instances in each class in earlier sections, ensuring the information is a good practice to minimize model error. As you can see from Figure 5, There are 2 classes in our dataset i.e. fire(1) and no fire(0). The distribution of class is 900 and 900 which are balanced. Thus there is no special treatment for the imbalance of the dataset here.

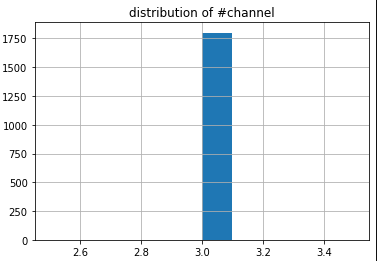
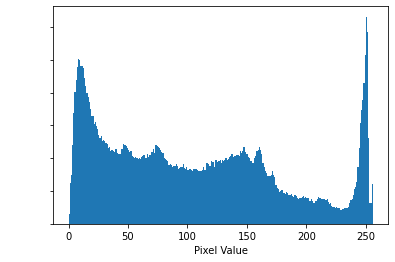
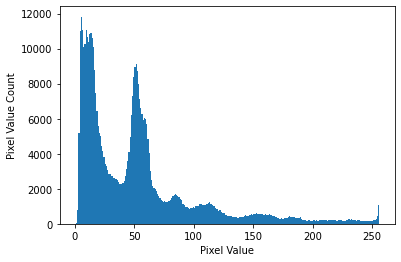


Figure 6: Distribution of channel number

Since we are concerned with image type i.e. grayscale, RGB or RGBA which have to be handled differently, number of channels is another aspect that needs to be investigated. From Figure 6 and Figure 2, we can conclude that all images have 3 channels i.e. RGB. So, all images can be used for our model.

In order to fix this issue we decided to plot grayscale histograms for each image and then average them. We did that for both the Fire class and Normal class.



It is not immediately obvious, but for some fire images tend to have more black in them than the neutral images. Our intuition was that fire would show up as white everywhere, but the reality might be that due to a very bright flame image cameras can focus on fire and get overexposed and make everything dark around the flame. In any case this information might be picked up by our models.

### **Issues**

* The dataset that we are using contains images of 3 different classes that are fire, smoke or neutral. We’ve left out the smoke images so that we can use this dataset to classify between fire and no-fire.
* The dataset may be too small to make models have high performance. However, the larger the dataset the more computationally expensive. The data may not fit in memory while training.
* There is class imbalance in additional dataset

### **Data Pre-processing**

**Preprocessing data for modeling**

* Images are resized to be 150x150 according to the evaluation from images metadata
* standardize data by using min/max normalization
* Images are splitted into training, validation and testing set as following
  + Training: 800 images
  + Validation: 100 images
  + Testing: 100 images
* Image are augmented to increase the size of data as follow
  + Training: 6000 images
  + Validation: 600 images
  + Testing: 600 images
* Image augmentation are performing by
  + rotation
  + shift images
  + vertical and horizontal flip
* Up sampling: We have trained our model more with additional data which has class imbalance. So, we have used upsampling technique to increase data of “fire class” which make the classes prior to be 50%

### **Modeling**

Two algorithms that we have used in our models are MLP and CNN. We have tried to tune our MLP model but the performance has never improved. So, we finally only focus on CNN. CNN models are trained in 3 ways i.e.

* First model: train with only main dataset
* Second model: train with only additional dataset
* Third model: train the first model more with additional dataset

During training, we have used Adam optimizer to help us achieve a high accuracy model. We also tried to prevent overfitting by setting the training to stop when there is no improvement of validation loss. Furthermore, we also use additional additive learning rate by monitoring the improvement of validation loss. The learning rate will reduce by a factor of 0.1 if the validation loss improves less than 0.0001 for 5 consecutive epochs.

### **Conclusion**

We have used two deep learning algorithms for training the model, MLP AND CNN. After evaluating both of the models the CNN performs better than MLP for our dataset. We have trained our model with two datasets and we have trained with 3 different parameterized models. The results obtained from the MLP were not as satisfactory as the results obtained from CNN. Using MLP we have obtained the accuracy of 0.71, precision of 0.75, recall of 0.66 and F1 of 0.70, which are not the great numbers. The evaluation numbers look good with the CNN training. With the predictions on the main dataset and training with CNN on the main dataset, the accuracy is 0.90, precision is 0.87, recall is 0.94, F1 is 0.91. With the predictions on the main dataset and training with CNN on additional dataset, the accuracy is 0.84, precision is 0.88, recall is 0.79, F1 is 0.83. With the prediction and re-training on the main dataset with CNN, the accuracy is 0.91, precision is 0.93, recall is 0.90, F1 is 0.91. With the predictions on the main dataset and training with CNN on the main dataset, the accuracy is 0.90, precision is 0.87, recall is 0.94, F1 is 0.91.

## 