Using Neural Networks and K-means Clustering for Accurate Wildfire Environmental Conditions Detection

Gurik Mangat

Abstract

In the field of fire dynamics, although a modest number of studies on wildfire analysis exist, there is a lack of implemented computational methods to accurately detect the presence of wildfire conditions. The need for these condition detection models is burgeoning as the substantial emissions of greenhouse gasses accelerate the rate of climate change. As a result of this, the severity and frequency of wildfires is drastically exacerbated year over year. In regards to this growing threat, my project aims to utilise artificial neural networks (ANNs) integrated into an application interface to make wildfire environmental conditions detection fast and accurate for firefighters in the field. In order to achieve this, we constructed a dataset containing a list of widely accepted environmental conditions that contribute to wildfire spread and ignition, and utilised a K-means clustering algorithm on NDVI imagery to analyse fuel moisture. Finally, this was all integrated into an easy-to-use desktop application (further work can be done to create a mobile version). This approach successfully determined the presence of dangerous wildfire environments in environmental data at an accuracy rate of over 98%. By giving firefighters the ability to use accurate and intelligent solutions, we aim to make the process of firefighting much safer, easier, and drastically more efficient.

1. Introduction

Over 67,000 wildfires and more than 7.0 million acres burned annually on average over the last 10 years in just the United States alone(K. Hoover & L. Hanson, 2019). In order to assess the risk of wildfires based on qualitative and environmental observations, many methods currently exist. Common examples of these include the Canadian Forest Fire Weather Index System, the National Fire Data System, and the National Fire Danger Rating System. However, the biggest struggle faced by these indices is analyzing and utilizing data in a practical and effective way(M. Hinds-Aldrich et al., 2017). Although some efforts have been made to modernize firefighting by organizations such as the US Geographical Service(USGS) when they developed the GeoMAC(USGS, 2005) system to digitally map and present current wildfire situations, the need for modern computational methods that can rapidly and accurately detect wildfires based on the conditions data still stands.

In current methods, input from the various fire danger rating systems mentioned above is evaluated by human firefighters who rely on past experiences and exposure to determine whether or not a region is at a higher risk of fire(United States Forestry Service, 1996). Although firefighter experience is an important aspect to firefighting, it is subject to common problems such as human error, lack of experience, and the fact that humans simply cannot process as much information as any other computerized system could. This study aims to solve that by using a binary classifier neural network trained on the current wildfire environmental conditions data and have it detect whether or not wildfires could be present in a certain area, as opposed to firefighters making subjective decisions.

Binary classification neural networks have been proven to learn and extract unique new interpretations from large amounts of data, and this makes them a suitable approach. The neural network developed in this study takes in 6 environmental conditions as inputs and outputs a value between 0 and 1, with 0 being wildfire conditions are not present and 1 being wildfire conditions are present.

2. Methods

The procedure of this project was split into four main steps, (1) determining which environmental conditions play a decisive role in wildfire environments so that they can be used as inputs for the neural network, (2) obtaining data and constructing a dataset using the chosen inputs, (3) training a classification neural network on the gathered dataset for accurate detection of wildfire conditions, and (4) integrating this trained machine learning model into an easy-to-use, prototype desktop interface with the possibility of completing a mobile-version which firefighters could use out on the field.

2.1 Environmental Conditions

There are a variety of qualitative wildfire condition ranking systems and indexes widely used by firefighting departments. Each of these includes information about what environmental conditions are considered as factors when it comes to judging whether or not a certain area is at a higher risk of fire. Through careful analysis of the indexes and systems listed in *1. Introduction*, the following environmental conditions were chosen as inputs for the neural network: temperature, humidity, wind speed, soil temperature, soil moisture, and vegetation health.

To effectively collect data about vegetation health in a certain region, Normalized Difference Vegetation Index (NDVI) imagery was used (figure 1). NDVI imagery provides a graphical representation of vegetation health by measuring the difference between near-infrared light(which vegetation reflects) and red light (which vegetation absorbs). Healthy vegetation that contains larger amounts of chlorophyll, water, and moisture has higher NDVI values. Those values are then colour-coded onto an image where the healthier vegetation (less prone to burning) shows up as green, while drier vegetation (more prone to burning) shows up as red. The following equation is used to calculate NDVI where NIR is near-infrared light and RED is red light: (1)

$$NDVI = \frac{[NIR - RED]}{[NIR + RED]}$$

In order for the dataset to be able to train the neural network, a K-means clustering algorithm was used to quantify color-coded information in an image so that it could be added to a dataset. K-means clustering is a technique that groups different observations into distinct clusters. The RGB (red, green, blue) values of pixels in the image are taken and assigned to a nearest cluster(figure 2). Each of those clusters represents a different colour and the largest cluster represents the most dominant colour. The center of each cluster (also known as the centroid) is located by averaging the distances of all the associated points. This is carried out by using the following distance equation where the *x* and *y* values are the coordinates of the points:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{2}$$

The centroid of each cluster gives us the RGB values of the most dominant color in the NDVI image. Due to the fact that fuel moisture and health levels are visually represented in NDVI imagery, knowing the dominant color allows us to determine whether or not the vegetation(fuel) in that image is dominantly dry if the colour has a higher R(red) value, or moist if the value is a lower R-value with more emphasis on the G(green) and B(blue) values. This approach allowed us to extract and quantify numerical insights into fuel moisture which were then included in the dataset.

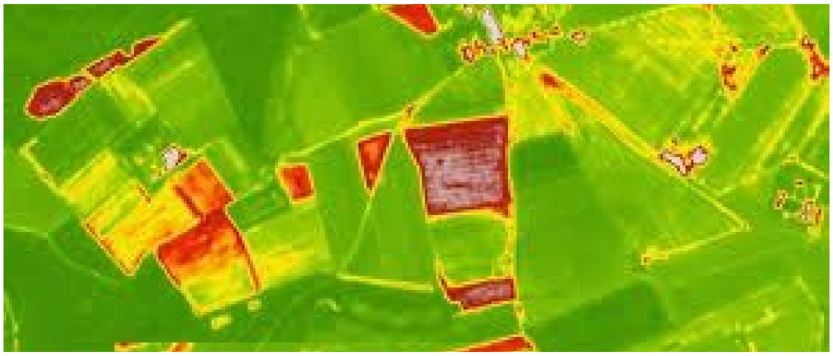


Figure 1: A sample NDVI image used for testing

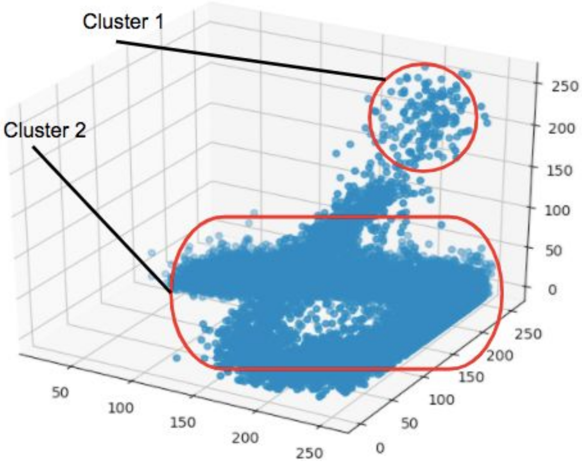


Figure 2: A graphical representation of the K-means clustering method used. Notice how “Cluster 2” is larger. The coordinates of the centroid of this will be the RGB values of the most dominant colour.

2.2 Dataset Construction

The following 6 environmental conditions were decided to be used as inputs for the neural network: temperature, humidity, wind speed, soil temperature, soil moisture, and K-means RGB values of the most dominant color from the NDVI imagery for vegetation health. The next step was to compile that information into a binary classification dataset that could be used to train the ANN.

To construct a dataset, the NASA FIRMS Active Wildfire Database was first used to locate a list of actively burning wildfires. Using this database, the geographical coordinates of those active fires were acquired, and environmental conditions data about their temperature, humidity, wind speed, soil moisture and soil temperature, was downloaded through the AgroDashboard API. This API was also used to acquire remote sensing high-resolution NDVI imagery taken by the Landsat-8 and Sentinel satellites that corresponded to each of those locations. Those images were then processed by the K-means algorithm to get the RGB values of their respective dominant colors. Numerous cases of fires and non-fires were collected and through several steps of data augmentation, the final dataset used for training contained around 2000 unique cases.

During dataset construction, this project also accounted for regional data bias by including balanced wildfire data from a multitude of different regions, terrains, and topographies from across North America. This is a distinct and crucial advantage over other wildfire prediction models as it exhibits the ability to dynamically generate accurate predictions even in foreign regions or conditions.

2.3 Artificial Neural Network

In order to begin generating wildfire detections, an Artificial Neural Network(ANN) was created. This network is a classifier, meaning it outputs a value between 1 and 0, with 1 being fire conditions are present, and 0 being fire conditions are not present. An example output would be 0.985. This would mean that the algorithm thinks there is a 98.5% chance of fire conditions being present. Conversely, a value of 0.23 would mean that there is only a 23% chance of fire conditions being present. In order to achieve high accuracy, if the output is at or above 0.95, then it is classified as fire conditions present.

The network architecture used in this study contains 1 input layer, 1 output layer, 2 hidden layers with 8 neurons each, 23 nodes, and 120 connections (figure 3). The Relu activation function was used due to its accelerated convergence of stochastic gradient descent, and its subsequently faster learning rate(A. Krizhevsky, I. Sutskever, & G.E. Hinton, 2017) as opposed to other activation functions such as sigmoid or tanh. For optimization, the Adam(Adaptive Moment Estimation) optimizer was used in order to update attributes of the network, such as weights, during training. The following is the function used by the Adam optimizer to calculate the weights in the neural network, where w_{t-1} are the model weights, eta(η) is the step size, the epsilon(ε) is to avoid a divide by zero error, and m_t, v_t are the estimators:

$$w_t = w_{t-1} - \eta \frac{\widehat{m}_t}{\sqrt{\widehat{v}_t} + \varepsilon} \tag{3}$$

For a loss function, binary cross-entropy was used due to the classification nature of this problem. Loss functions are used to optimize the parameter values in the model and measure the error between the network's output and the desired correct output. The following is the function used by the binary cross-entropy function where y is the binary indicator (0 or 1) for the test case, and p is the predicted probability (between 0 and 1) generated by the network :

$$-(y log(p) + (1-y) log(1-p)) \tag{4}$$

Notice how this function uses a logarithm. As the predicted probability of the network slowly reaches the correct value while training, log loss gradually decreases. However, as the predicted probability decreases, the log loss increases rapidly so that both types of errors are penalized, but especially ones where the prediction is wrong and the confidence is high. This is a marked advantage as this loss function leads to a better trained and robust model.

To split the dataset for training, it underwent a 70:20:10 split where 70% of the dataset was used for training, 20% was used for validation, and the last 10% was used for testing purposes. This ratio is slightly different from the commonly used Pareto principle which calls for a 80:20 split, but provides major benefits when training on relatively smaller sized datasets like the one used in this study because of the fact that it allows model validation to happen with greater accuracy on its different hyperparameters.

We deployed a dropout function with rate 0.20 after the first hidden layer to prevent neurons from excessively co-adapting. This method significantly reduces overfitting and is a major improvement over other regularization methods(N. Srivastava, G.E. Hinton, A. Krizhevsky, I. Sutskever, & R. Salakhutdinov, 2014).

The next hyperparameter which played a critical role was batch size. The batch size regulates the accuracy of the estimated error gradient which assigns the weights to connections while training, and this in turn influences the speed and stability of the learning process. We chose a batch size of 9 for training this model.

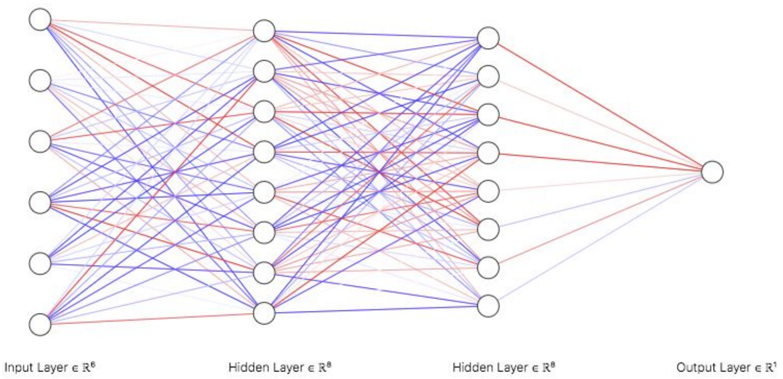


Figure 3: A diagram of the neural network architecture where each circle is a neuron, and the coloured lines in between them are connections with their weights.

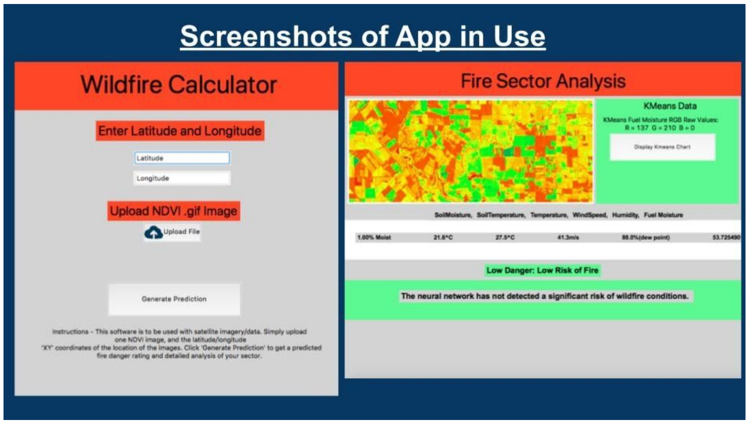
2.4 Application Interface

In order for firefighters to start making predictions, a desktop app (with potential to be mobile) was built on the Tkinter framework to allow them to utilise the algorithm to generate their own predictions with unique field-centric data (figure 4, figure 5, figure 6). The app works by first prompting operators to upload their coordinate locations. Those coordinates are then used to collect real-time, accurate weather data on humidity, wind speed, and temperature using the OpenWeatherMap API. The second thing those coordinates are used for, are to get data on soil moisture and soil temperature. The app uses a geometric-coordinate algorithm to find the coordinates of a virtual one-hectare square using those original coordinates. The coordinates of that one-hectare square are then fed into the OpenWeatherMap agriculture API. This API allows us to call accurate, real-time soil moisture and soil temperature data acquired by remote sensing sensors on the Landsat-8 and Sentinel satellites. This method is identical to how data was collected for the original training dataset, except instead of collecting data for multiple cases, it now collects real-time data from the firefighter's coordinates.

The second thing that firefighters have to upload is an NDVI image that corresponds to their input coordinates. After uploading that image, the K-means clustering algorithm is run on it. This gives us the RGB (Red, Green, Blue) values of the most dominant color in the image, which in turn, can give us insights into the fuel's moisture content.

After uploading the coordinate and image, the firefighter clicks the "Generate Prediction" button. This causes all the data that the app has collected to be fed into the neural network which then generates a prediction on whether or not fire conditions are present.

After the network generates a prediction, a detailed analytics page is shown. This page tells the user what the neural network's prediction is, along with the K-means fuel moisture and environmental data that was fed into it. Not only does this page give firefighters the prediction, but it also tells them what data was fed into the algorithm so that they themselves have a greater understanding of what is going on in their surroundings.


Screenshots of App in Use

Instructions - This software is to be used with satellite imagery/data. Simply upload one NDVI image, and the latitude/longitude 'XY' coordinates of the location of the images. Click 'Generate Prediction' to get a predicted fire danger rating and detailed analysis of your sector.

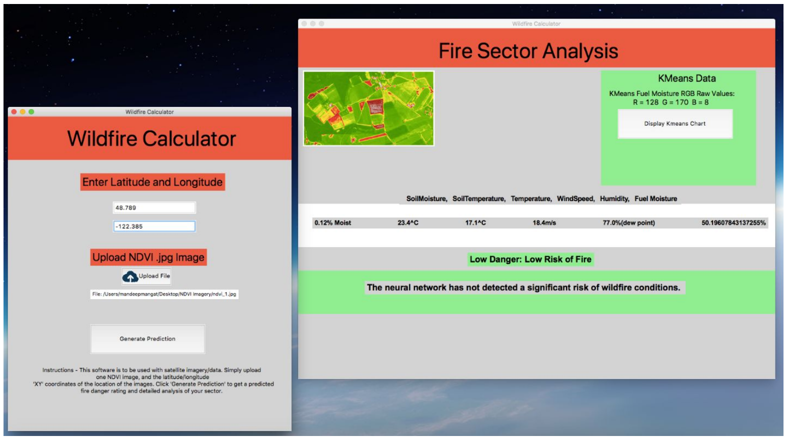
KMeans Fuel Moisture RGB Raw Values:
R = 137 G = 210 B = 0

```
1.00% Moist    21.6^C    27.5^C    41.3m/s    88.0%(dew point)    53.725490
```

Low Danger: Low Risk of Fire

The neural network has not detected a significant risk of wildfire conditions.

Figure 4: Screenshots of the application in use. The first window is the data entry pane in which the firefighter uploads his coordinates and a corresponding NDVI image. The second window is produced after clicking the "Generate Prediction" button and displays the environmental conditions, fuel moisture data in the form of K-means dominant colour RGB values, and the neural networks prediction.



48.789
-122.385

File: /Users/mandeepmangat/Desktop/NDVI Imagery/ndvi_1.jpg

Instructions - This software is to be used with satellite imagery/data. Simply upload one NDVI image, and the latitude/longitude 'XY' coordinates of the location of the images. Click 'Generate Prediction' to get a predicted fire danger rating and detailed analysis of your sector.

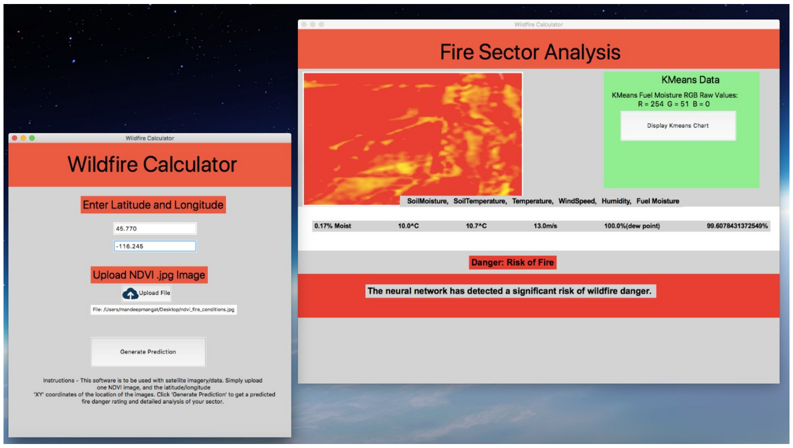
KMeans Fuel Moisture RGB Raw Values:
R = 128 G = 170 B = 8

```
0.12% Moist    23.4^C    17.1^C    18.4m/s    77.0%(dew point)    50.19607843137255%
```

Low Danger: Low Risk of Fire

The neural network has not detected a significant risk of wildfire conditions.

Figure 5: An example of when the app does not detect fire conditions.



45.770
-116.245

File: /Users/mandeepmangat/Desktop/ndvi_fire_conditions.jpg

Instructions - This software is to be used with satellite imagery/data. Simply upload one NDVI image, and the latitude/longitude 'XY' coordinates of the location of the images. Click 'Generate Prediction' to get a predicted fire danger rating and detailed analysis of your sector.

KMeans Fuel Moisture RGB Raw Values:
R = 254 G = 51 B = 0

```
0.17% Moist    10.0^C    10.7^C    13.0m/s    100.0%(dew point)    99.6078431372549%
```

Danger: Risk of Fire

The neural network has detected a significant risk of wildfire danger.

Figure 6: An example of when the app does detect fire conditions.

3. Results

3.1 Neural Network Training

To ensure the accuracy and robustness of the neural network, a number of different metrics such as training accuracy, validation accuracy, training loss, and validation loss were used. The training accuracy shows how the model is progressing as it is learning, while the validation accuracy gives a measure of the quality of the model based on the validation set every epoch. Loss functions on the other hand are used to optimize the parameter values in the model and measure the error between the network's output and the desired correct output. The training loss is the error the network makes while training, and the validation loss is the error the network makes while validating.

The first original, unoptimized model (figure 7) followed the conventional Pareto training validation split of 80:20, did not have a dropout layer, and had a batch size of 100. If you look at the Loss graph produced by the network after training (figure 7), you can see that this model shows large signs of overfitting. Overfitting is when the model has memorized the training examples and has not learned to generalize to new situations yet. This model also had a validation accuracy of 98.33%, which can be slightly improved upon further.

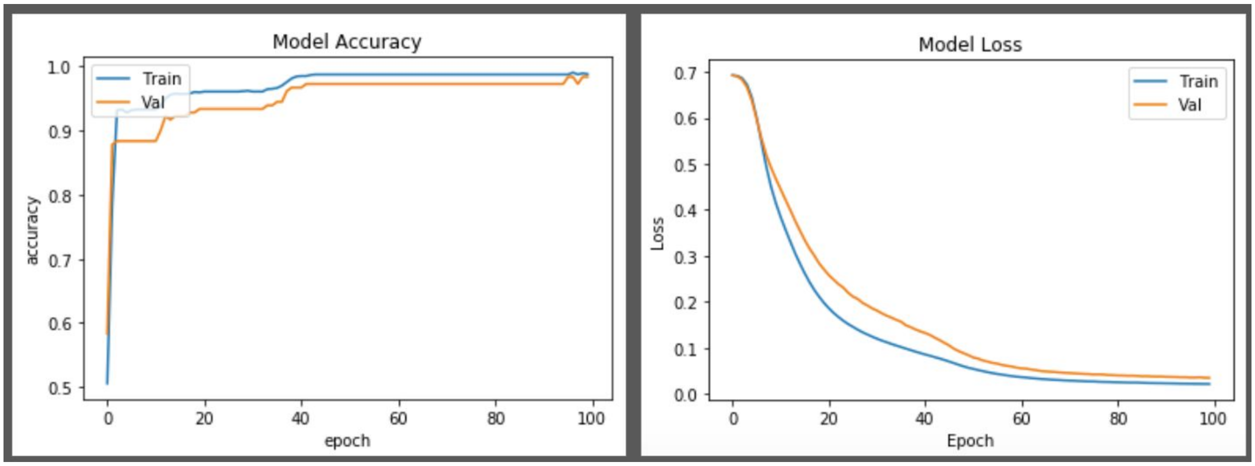


Figure 7: These are the Accuracy and Loss graphs produced by the original network. As you can see, this model displays large signs of overfitting when the Validation loss is greater than the Training loss in the Model Loss graph. This model also trained in a less consistent manner, and achieved an accuracy rate of 98.33% which can be improved upon further.

The next model uses the fine-tuned hyperparameters as discussed in "*2.3 Artificial Neural Network*" and leverages all the advantages mentioned in that section. It contains a dropout layer with rate 0.20 after the first hidden layer to account for the overfitting, a training:validation:test split of 70:20:10, and a batch size of 9. If you look at the Accuracy and Loss graphs produced by this model after training (figure 8), you'll see that the Accuracy loss and Validation loss plots both nicely converge, which shows us that there are virtually no signs of overfit. We also achieved a Validation accuracy of 98.61% as opposed to 98.33%, and we can see by looking at the Accuracy graph (figure 8), that this model trained in a much more stable and less volatile manner.

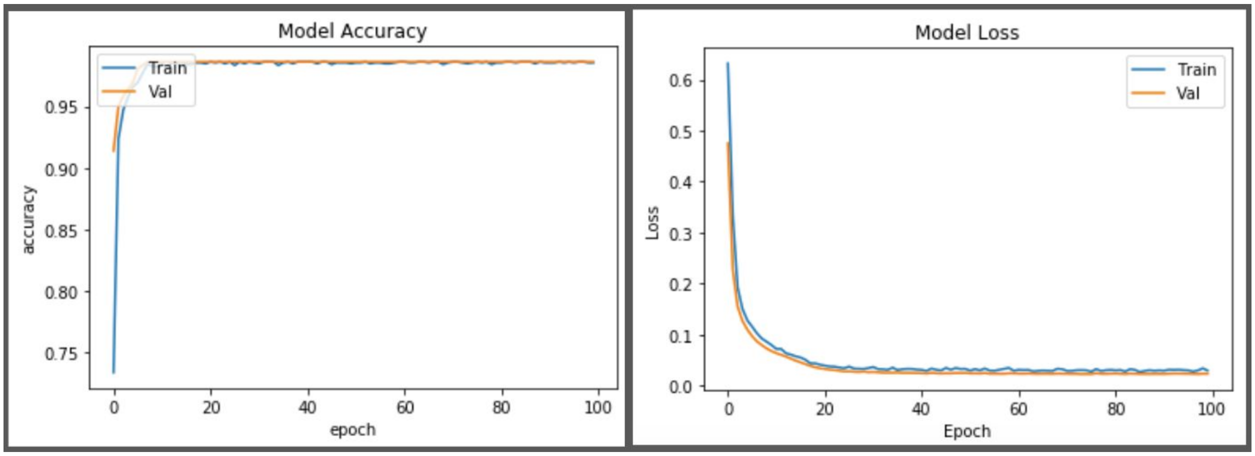


Figure 8: Accuracy and Loss graphs produced by the final fine-tuned network. As you can see, this model achieved a higher accuracy rate of 98.61%, contains virtually no overfitting, and was trained in a more stable and less volatile manner.

3.2 Completed Model Testing

After training was completed and a final model was produced, an automated test (figure 7)was run on the saved model in order to determine the final experimental accuracy. We used our test set that was kept aside for testing during the initial 70:20:10 data split. This test set had 200 unique cases and the model was able to correctly classify 196 of them as either “Fire Conditions Present”, or “Fire Conditions Not Present”. This equates to a final accuracy of 98%.

4. Discussion

The primary purpose of this study was to develop a neural network that could accurately detect wildfire conditions using environmental factors as inputs. Earlier research(Y.O. Sayad, H. Mousannif, & H.A. Moatassime, (2019)) used a similar approach by using NDVI imagery and LST (Land Surface Temperature) measurements as inputs for an artificial neural network. However for this study LST was not used and instead, we opted for a more wider array of environmental conditions that would allow for greater diversity and breadth in the input data.

Furthermore, another approach(S.R. Coffield et al, 2019) utilised neural networks to predict the final size of wildfires after ignition. Although final fire size prediction after ignition was not the focus of this study, it is seen as an exciting extra future addition.

No study to our knowledge however used K-means clustering as a method of fuel moisture quantification. Although this study achieved high levels of accuracy, subsequent research could be conducted to go more in-depth into its potential.

5. Conclusion

With this research, artificial neural networks (ANNs) were successfully constructed and trained to classify regions on whether or not wildfire conditions were present based on environmental conditions inputs. Relying on current methods, firefighters must rely on inaccurate and highly subjective human inference and experience. However, by using a myriad of environmental conditions based on the Canadian Forest Fire Weather Index System, and by using new innovative techniques such as k-means clustering for fuel moisture quantification on NDVI imagery, a dataset was created which was used to train an artificial neural network. By taking advantage of highly optimized hyperparameters and neural network architectures, this model achieved an accuracy of 98% during testing. This has successfully demonstrated the effectiveness of our technique and will provide major imperative leaps to the fields of fire dynamics, sustainable forestry management, and wildfire danger mitigation.

References

Government Reports

Hoover, K., & Hanson, L. A. (2019). *Wildfire statistics*. Congressional Research Service. https://fas.org/sgp/crs/misc/IF10244.pdf

Hinds-Aldrich, M., Dr., Knight, M., Nicolosi, A., & Evarts, B. (2017). *National Fire Data Survey: Findings on the State of the Existing American Fire Data Ecosystem*. National Fire Protection Association.
https://www.nfpa.org/~/media/96AD4BAEA2684EE3B2E3F73D80D1B60E.pdf

USGS. (2005). *Wildfire Hazards—A National Threat*. United States Geographical Survey. https://pubs.usgs.gov/fs/2006/3015/2006-3015.pdf

United States Forestry Service. (1996). *Wildland Firefighter Safety Awareness Study*. TriData. https://www.nifc.gov/safety/safety_documents/phase1.pdf

Journal Research Papers

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, *60*(6), 84-90. https://doi.org/10.1145/3065386

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Dropout: A Simple Way to Prevent Neural Networks from Overfitting, 15*(56), 1929-1958.

Sayad, Y. O., Mousannif, H., & Al Moatassime, H. (2019). Predictive modeling of wildfires: A new dataset and machine learning approach. *Fire Safety Journal*, *104*, 130-146. https://doi.org/10.1016/j.firesaf.2019.01.006

Coffield, S. R., Graff, C. A., Chen, Y., Smyth, P., Foufoula-Georgiou, E., & Randerson, J. T. (2019). Machine learning to predict final fire size at the time of ignition. *International Journal of Wildland Fire*, *28*(11), 861. https://doi.org/10.1071/wf19023