Memo

To:	Professor Pisano
From:	Rishav De (rishavde@bu.edu), Pranet Sharma(pranetsh@bu.edu), Jueun Kang(vkang@bu.edu), Thomas Bowler(tsbowler@bu.edu), Young-Chan Cho(yccho@bu.edu)
Team:	Team 1
Date:	4/4/2024
Subject:	Team 1 Final Project Testing Report

1.0 Equipment

FireNet Model

Hardware:

- Computer: A machine with CPU, GPU, and storage for deep learning model training and testing
- Internet Connection: For downloading large datasets and accessing cloud resources

Software:

- Python
    - TensorFlow / Keras: For building and training deep learning models.
    - OpenCV: For image processing.
    - Matplotlib / sklearn: For data visualization and performance evaluation.
- Google Colaboratory
    - For running models in .ipynb files using GPU

Binary Classification Model

Hardware:

- Computer: A machine with CPU, GPU, and storage for deep learning model training and testing
- Internet Connection: For downloading large datasets and accessing cloud resources

Software:

- Python
    - Tensorflow
    - Image Augmentation and Generation
    - Keras
    - Numpy, Scipy
    - Plotly
    - sklearn
    - seaborn
- Google Colaboratory
    - For running models in .ipynb files using GPU

FireDataUS Model

Hardware:

- Computer: A machine with CPU, GPU, and storage for deep learning model training and testing
- Internet Connection: For downloading large datasets and accessing cloud resources

Software:

- Python
  - Tensorflow
  - Image Augmentation and Generation
  - Keras
  - Numpy, Scipy
  - Plotly
  - sklearn
  - seaborn
  - sqlite3
- Google Colaboratory
  - For running models in .ipynb files using GPU

Website

Hardware:

- Computer: A machine to view the website on
- Internet Connection: For accessing the website

2.0 Setup

2.1 Software Installation and Configuration:

I. Python Installation

II. Library Installations: Install TensorFlow and Keras, the primary libraries for building and training your deep learning model. TensorFlow offers a comprehensive set of tools and libraries for machine learning, while Keras simplifies neural network configurations.

III. OpenCV for Image Processing: Install OpenCV, which allows you to manipulate images and prepare them for modeling, such as resizing and reshaping.

IV. Setting up Google Colab: Google Colab provides powerful computing resources that could be used for processing large datasets and speeding up model training.

V. Data Visualization Tools: Install Matplotlib and sklearn. Matplotlib is necessary for plotting graphs and visualizing data, which is vital for analyzing model performance. Sklearn offers data mining and analysis tools, including performance metrics like confusion matrix.

2.2 Google Drive and Dataset Setup:

I. Google Drive: This is where you'll store your datasets and trained models.

II. Connecting to Google Colab: In your Python scripts, include the code to mount your Google Drive to Google Colab. This step ensures that Colab can access and use the data stored in your Drive.

III. Uploading Dataset: Upload your training and testing datasets to a specific folder in Google Drive. Organize your data in a way that is easily accessible.

3.0 Testing Procedure

FireNet Model

I. Model Loading and Initialization: Start by loading the pre-trained model using Keras.

II. Testing Data Preparation: Use the same function as in training to prepare the test dataset, ensuring it's separate from the training and validation sets. Apply consistent preprocessing, including resizing and normalization.

III. Model Prediction: Run predictions on the test data using the model.predict() function, and store these predictions for analysis.

IV. Performance Evaluation: Construct a confusion matrix from the predictions and actual labels. Calculate key performance metrics such as Accuracy, Precision, Recall, and F-Measure to quantitatively assess the model.

V. Visualization and Analysis: Utilize Matplotlib for visualizing the confusion matrix and other metrics. Conduct an in-depth analysis to identify the model's strengths and areas needing improvement, focusing on biases or misclassifications.

VI. Documentation and Reporting: Document all findings, including performance metrics and insights. Provide clear reporting on the model’s performance and suggest potential improvements or future modifications.

Binary Classification Model

I. Dataset Loading: Start by loading the dataset from Google Drive
II. Dataset Analysis: Observe dataset balance and load into a data frame for further use in model
III. Image Augmentation and Generation: Generate Train, Validation and Test sets for model
IV. Model Creating: Define model and add layers and parameters
V. Model Compile and Fit: Compile and fit model using certain metrics and callbacks
VI. Model Evaluate: Evaluate the model using test data on both dataset sizes
VII. Visualization and Analysis: Use confusion matrices and single image test to show model output and analyze areas of improvement.

FireDataUS Model

I. Model Loading and Initialization: Start by loading the pre-trained model using Keras.
II. Testing Data Preparation: Use the same function as in training to prepare the test dataset, ensuring it's separate from the training and validation sets. Apply consistent preprocessing, including resizing and normalization.
III. Model Prediction: Run predictions on the test data using the model.predict() function, and store these predictions for analysis.
IV. Performance Evaluation: Construct a confusion matrix from the predictions and actual labels. Calculate key performance metrics such as Accuracy, Precision, Recall, and F1-Score to quantitatively assess the model.
V. Visualization and Analysis: Utilize Matplotlib for visualizing the confusion matrix and other metrics. Conduct an in-depth analysis to identify the model's strengths and areas needing improvement, focusing on biases or misclassifications.
VI. Documentation and Reporting: Document all findings, including performance metrics and insights. Provide clear reporting on the model's performance and suggest potential improvements or future modifications.

Website

I. Website Loading: Visit the website at the URL sigma31.github.io/ec464-website
II. Website Navigation: Navigate to the various different pages on the website and view the content.

4.0 Measurements

FireNet Model

i. Precision: The ratio of true positive samples among all samples predicted as positive.
   1. TP / (TP + FP), where TP is True Positives and FP is False Positives.

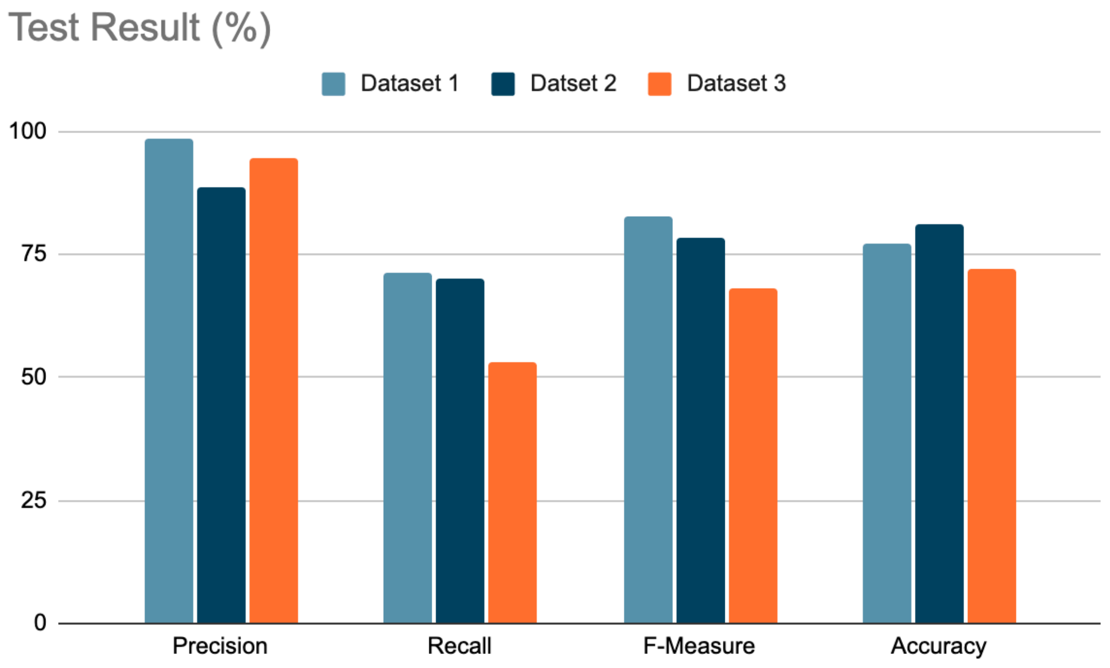
ii. Recall: The ratio of true positive samples detected among all actual positive samples.
   1. TP / (TP + FN), where FN is False Negatives

iii. F-Measure: The harmonic mean of Precision and Recall.
   1. 2 * (Precision * Recall) / (Precision + Recall)

iv. Accuracy: The percentage of samples correctly predicted by the model.
   1. (TP + TN) / Total number of cases

	Dataset 1	**Correct**	**Dataset 2**	**Correct**	**Dataset 3**	**Correct**
Precision	98.67%	Yes	88.52%	Yes	94.44%	Yes
Recall	71.37%	Yes	70.08%	Yes	53.13%	No
F-Measure	82.83%	Yes	78.22%	Yes	68.00%	No
Accuracy	77.26%	Yes	81.07%	Yes	71.93%	Yes

* Correct: >70%



- Dataset 1: FIRE Dataset
    - https://www.kaggle.com/datasets/phylake1337/fire-dataset
- Dataset 2: Fire Images Dataset
    - https://www.kaggle.com/datasets/rachadlakis/firedataset-jpg-224
- Dataset 3: Forest Fire Images
    - https://www.kaggle.com/datasets/mohnishsaiprasad/forest-fire-images

Binary Classification Model

I. Precision: The ratio of true positive samples among all samples predicted as positive.
    A. TP / (TP + FP), where TP is True Positives and FP is False Positives.

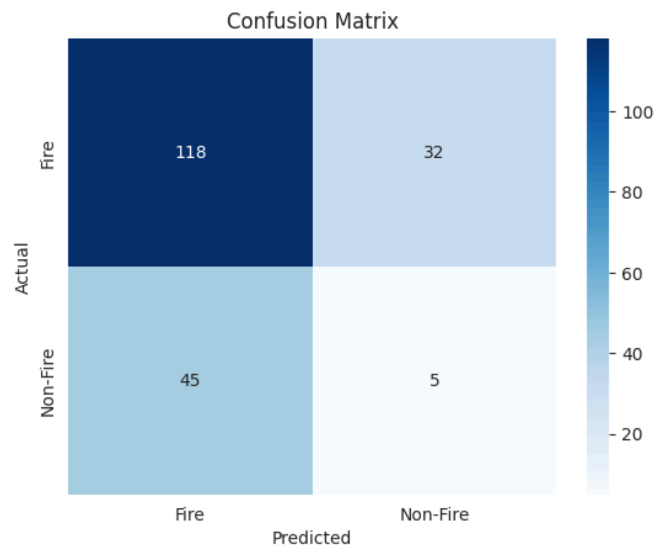
II. Recall: The ratio of true positive samples detected among all actual positive samples.
    A. TP / (TP + FN), where FN is False Negatives

III. F1-Score: The harmonic mean of Precision and Recall.
    A. 2 * (Precision * Recall) / (Precision + Recall)

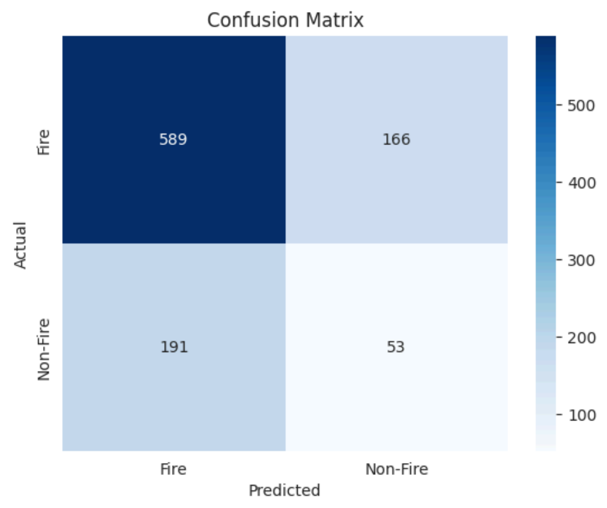
IV. Accuracy: The percentage of samples correctly predicted by the model.
    A. (TP + TN) / Total number of cases

These criteria have been selected to better understand how well the model is performing when compared to similar models.

	Prediction DS 1	**Prediction DS 2**	**Correct (Y/N**
Precision	Fire >0.70 Non-Fire >0.10	Fire >0.75 Non-Fire >0.20	Y Y
Recall	Fire >0.70 Non-fire >0.10	Fire >0.75 Non-fire >0.20	Y Y
F1-Score	Fire >0.70 Non-fire >0.10	Fire >0.75 Non-fire >0.20	Y Y
Accuracy	>0.60	>0.60	Y



Prediction DS 1



Prediction DS 2

FireDataUS Model

1. Precision: The ratio of true positive samples among all samples predicted as positive.
    a. TP / (TP + FP), where TP is True Positives and FP is False Positives.
2. Recall: The ratio of true positive samples detected among all actual positive samples.
    a. TP / (TP + FN), where FN is False Negatives
3. F1-Score: The harmonic mean of Precision and Recall.
    a. 2 * (Precision * Recall) / (Precision + Recall)
4. Accuracy: The percentage of samples correctly predicted by the model.
    a. (TP + TN) / Total number of cases

These criteria have been selected to better understand how well the model is performing when compared to similar models.

Different Fire classes will have different scores. Since data is heavily imbalanced, looking only at class A and B.

	Prediction	**Correct (Y/N)**
Precision	A >0.70 B >0.60	**Y**
Recall	A >0.70 B >0.60	**Y**
F1-Score	A >0.70 B >0.60	**Y**
Accuracy	>0.60	**Y**

Dataset - https://www.fs.usda.gov/rds/archive/catalog/RDS-2013-0009.6

Website

The following criteria are meant to serve as a measure of how easy to use and accessible the website is:

Feature	**Correctly Working (Y/N)**
The website homepage loads when the website is visited.	**Y**
The navigation header works correctly for the web page.	**Y**
All images and content correctly load and are displayed.	**Y**
The scroll functionality works correctly for the web page.	**Y**
All the embedded links work correctly and none of the links are broken.	**Y**

5.0 Conclusion

The following conclusions can be derived based on the functional requirements of the project:

- Each model provides a distinct benefit and can be tailored to specific use cases. These distinct benefits are discussed below.
- The FireNet model provides the best overall accuracy and is suitable for cases where precision and accuracy are of utmost importance and computational resources are not limited. Future directions for improvements to the model are suggested on the website for any researchers interested in further tuning the model according to their own use case. One such suggestion is applying threshold tuning.
- The Binary Classification model is most suitable for cases where computational resources are limited. It provides good accuracy and precision while utilizing minimal computational resources. This model is most suitable for an early alarm system that is deployed on the edge on embedded devices. Future directions for researchers interested in this model would involve tailoring it to whatever particular platform they wish to deploy it on.
- The FireDataUS model is a unique model that uses text data instead of image data like the other models. This model is unique as it leverages historical data in order to predict fire sizes and can be used to augment traditional wildfire detection AI models.
- The web page serves as the central repository for all the information regarding the topic of wildfire detection. On the web page users can easily view the implementation of all three models and download them. They can also read about the theoretical reasoning behind each model's development and download the datasets used for each model. The web page also provides additional resources in the form of datasets that can be leveraged by those interested in the problem of early wildfire detection.