Model and Documentation Details

Model File

- The CNN model file, due to its large size, is hosted on Google Drive. It can be accessed at - *https://drive.google.com/file/d/1_xfvZ66aV3vk9MgshKraHesZI50ScXn1/view?usp=sharing*

Model Training Data

- The training dataset has been optimized to approximately 10,500 files to reduce training time and enhance modeling effectiveness by looking at the system configuration it was being modelled on.

Deployment Documentation

- **Deployment Screenshot**: A sample screenshot of the model deployment on localhost is saved as **[Deployment]_Flask_LocalHost.png**.
- **Deployment Script**: The Python script for Flask deployment is available in **[Deployment]_Flask_LocalHost.py**.
- **Deployment Script Explanation**: A detailed explanation of the deployment script is documented in **[Deployment]_Script_Explanation.pdf.**

Prediction Documentation

- **Prediction Script**: The Python script used for making predictions with the CNN model is named **[Prediction]_Predict_CNN.py**.
- **Prediction Script Explanation**: The explanation for the prediction script is provided in **[Prediction]_Script_Explanation.pdf**.

Training and Validation Documentation

- **Training and Validation Script**: The Python code for training and validation is found in **[Training]_Train_Valid_CNN.py**.
- **Training and Validation Plot**: An image showing the plot of training versus validation accuracy across epochs is saved as **[Training]_Train_Valid_Plot.png**.
- **Training Run Results**: Compilation results from running the training and validation script for all 7 epochs are documented in **[Training]_Train_Valid_Run.txt**.
- **Hyperparameter Tuning Documentation**: Information about manual hyperparameter tuning utilized in the model is in **Hyperpara.pdf**.

Detailed Overview

1. Design Choices

- **Model Architecture**: The CNN model comprises three convolutional layers each followed by max-pooling layers, a flattening step, and two dense layers. The final layer uses softmax activation to categorize the images into three classes: fire, no fire, and smoke.
- **Data Augmentation**: Techniques such as rotation, width and height shifts, shear, zoom, and horizontal flips are implemented to enhance the model's generalization and minimize overfitting.

2. Performance Evaluation

- **Training Regime**: The model was trained over 7 epochs with a batch size of 32, including a validation subset to track performance on unseen data.
- **Results**: The model achieved over 90% accuracy on both the training set and the test set, though validation accuracy was slightly lower, suggesting mild overfitting.

3. Future Work

- **Network Architecture**: Exploration of deeper and more complex architectures such as ResNet or Inception could be considered.
- **Data Augmentation and Hyperparameters**: Further adjustments in data augmentation and an extensive hyperparameter tuning phase could potentially elevate model accuracy and robustness.
- **Dataset Expansion**: Enlarging the dataset might help in improving the model's ability to generalize.