For-rest from Fires

Project Overview

This project aims to develop a fire detection system using deep learning techniques. Traditional fire detection systems relying on temperature or smoke sensors have limitations such as slow response times and inefficiency when the fire is distant from the detectors. This project explores computer vision and image processing techniques as a cost-effective alternative.

Objectives

Early fire detection is vital for averting disasters and saving lives. Conventional fire detection methods, like smoke detectors and human observation, can be slow and ineffective in certain situations. This project utilizes deep learning techniques to build a CNN model that can automatically detect fire, smoke, and non-fire in images, offering a quicker and more dependable solution. The main goal of this project is to create a fire detection system that can identify fires from images obtained from surveillance systems or other resources. The key objectives include:

- Collecting data on different types of fires, including images and videos of flames, smoke, and heat.
- Preprocessing the data to ensure it is ready for training the deep learning model.
- Developing a deep learning model using convolutional neural networks (CNNs) to detect fires in the collected data.
- Evaluating the model's accuracy using a test set.

Methodology

Data Extraction: Zip File Handling: The dataset, stored in a zip file, is extracted to a designated directory using the zipfile module. Directory Structure: The dataset is arranged into training and testing directories, facilitating the data loading process.

Data Augmentation: ImageDataGenerator: The ImageDataGenerator class from tensorflow.keras.preprocessing.image is employed for data augmentation, applying random transformations such as rotation, width/height shifts, shear, zoom, and horizontal flip to the training set to improve model generalization.

Model Architecture: Pre-trained Model: A pre-trained MobileNetV2 model (with ImageNet weights) serves as the base model, utilizing transfer learning. Custom Layers: The model incorporates a GlobalAveragePooling2D layer, a Dense layer with 128 units and ReLU activation, and a final Dense layer with 3 units and softmax activation for classification.

Learning Rate Scheduler: LearningRateScheduler Callback: A custom learning rate scheduler is employed to reduce the learning rate after a specified number of epochs, aiding in fine-tuning the model and mitigating overfitting.

Data Collection

Collect images and videos of flames, smoke, and heat in a controlled environment to ensure the data is representative of real-world fires.

Data Preprocessing

Resizing: All images are resized to a consistent dimension to be compatible with the CNN. Normalization: Pixel values are scaled to a range between 0 and 1. Data Augmentation: Techniques like rotation, flipping, and zooming are applied to increase the variety within the training dataset.

Prediction and Visualization

Image Preprocessing: User-provided images are loaded, resized, and pre-processed before being input into the model for prediction. Result Display: The predicted class labels are shown alongside the input images using matplotlib. Confusion Matrix and Accuracy: A confusion matrix is created to visualize the model's performance across different classes, and the overall accuracy score is calculated and displayed.

References

The project necessitates the following reference: Python 3.x Google Colab TensorFlow NumPy Pandas Matplotlib scikit-learn

```
from google.colab import drive

import zipfile

import os

import tensorflow as tf
```