Documentation for Wildfire Detection

Gema Csanád

Project Introduction

For this project, I wanted to combine image classification with an environmental cause, which led me to focus on wildfire detection using satellite imagery.

Wildfires are a significant environmental threat, leading to the destruction of forests, wildlife habitats, and even residential areas. Timely detection is crucial for minimizing damage. However, current wildfire monitoring methods often rely on ground reports and manual satellite image analysis, both of which can be slow and inefficient.

In this project, I aim to develop a deep learning model capable of classifying satellite images as either "wildfire" or "no wildfire." The goal is to support faster and more accurate detection, ultimately contributing to more effective wildfire response efforts.

Previous Solutions

- **Landsat-8 Active Fire Detection**: Trained on dataset comprising over 150,000 image patches from Landsat-8 imagery. They evaluated various convolutional neural network (CNN) architectures for active fire detection, achieving a precision of 87.2% and recall of 92.4%. https://arxiv.org/abs/2101.03409?context=cs.CV
- **3D CNN for GOES-16 Data**: A study proposed a 3D CNN model that integrates spatial and spectral information from GOES-16 satellite images, achieving an F1-score of 94% in wildfire detection tasks. https://www.mdpi.com/2571-6255/6/5/192

Dataset

For this project, I selected a dataset from Kaggle containing satellite images of Canada, labeled as either *wildfire* or *no wildfire*. The dataset includes approximately 43,000 images, with an even distribution between the two classes. It is split into 70% for training, 15% for validation, and 15% for testing.

Before settling on this dataset, I explored several alternatives. However, most of the other datasets I found were either not based on satellite imagery, differed significantly in format or quality, or lacked proper labeling.

Fortunately, the chosen dataset is clean and well-structured: all images are labeled correctly, there are no duplicates, and the class distribution is balanced. This makes it a solid foundation for training an effective deep learning model.

Data Augmentation

To improve the model's ability to generalize, I applied several data augmentation techniques to the training images:

- **Rotation**: Images were rotated by 30 degrees to simulate different orientations.
- **Random Horizontal Flipping**: Applied with a 50% probability to introduce spatial variability.
- **Resizing:** Standardized image sizes to ensure robustness across different resolutions.
- **Color Adjustments:** Changes in brightness, contrast, and saturation were used to simulate varying lighting conditions.

One of the early challenges I encountered was distinguishing smoke from clouds, which I addressed by training the model to learn differences in texture and brightness using a 2D convolutional approach.

Model Choice

For this project, I chose to build a 2D Convolutional Neural Network (CNN) because it aligns well with the spatial and visual characteristics of satellite imagery. Here's why:

- **Spatial Feature Extraction**: Wildfires exhibit patterns (plumes, burn marks, and vegetation changes) that have distinct spatial structures. 2D CNNs are designed to capture these spatial hierarchies effectively.
- **Translation Invariance**: A fire can occur anywhere in an image. CNNs can detect relevant features regardless of their position, thanks to convolution and pooling layers.
- **Hierarchical Learning**: The network learns low-level features (like edges and textures) in early layers and more complex patterns (like full smoke plumes or burn areas) in deeper layers.
- **Multi-Channel Compatibility**: The dataset contains RGB images, and 2D CNNs are well-suited for processing such multi-channel input by learning color relationships and spatial patterns simultaneously.
- **Proven Effectiveness**: 2D CNNs are the standard in image classification tasks, with many robust architectures and tools available to support their implementation and optimization.
- **Distinguishing Subtle Features**: Even though differentiating smoke from clouds is challenging, the CNN can learn subtle texture and brightness cues that help distinguish between them.

Baseline Model

To begin, I implemented a small baseline CNN model to better understand the data and verify that the overall pipeline works correctly. Small models help avoid overfitting early on and often generalize better.

This initial model contains approximately 63,617 parameters, which is relatively lightweight for a CNN. Despite its simplicity, it provides a useful benchmark and allows me to quickly test preprocessing steps, augmentation techniques, and training behavior.

For the activation functions:

- I used **ReLU** (Rectified Linear Unit) in the hidden layers because of its non-linearity, which enables the model to learn complex patterns. It's also widely used in CNNs due to its efficiency and performance.
- The final layer uses a **sigmoid** activation function, which maps the output to a probability between 0 and 1—ideal for binary classification tasks like wildfire vs. no-wildfire.

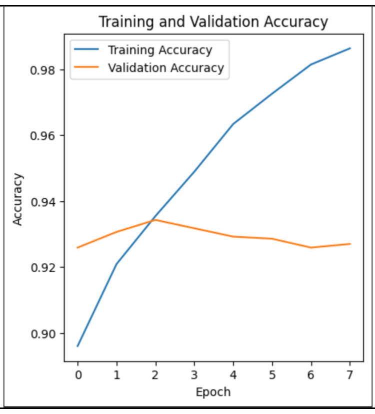
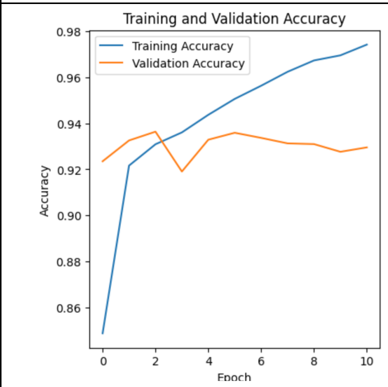
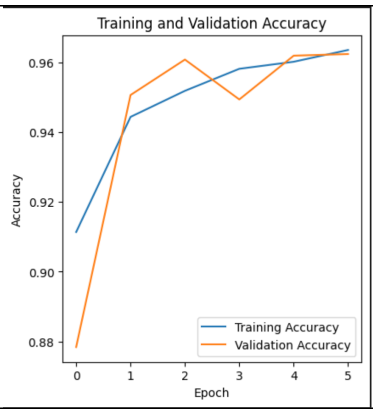
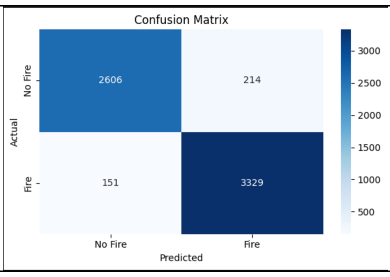
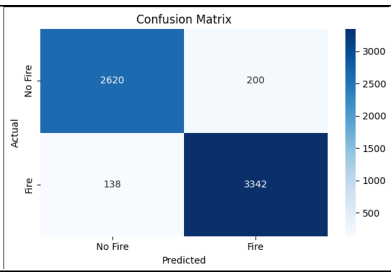
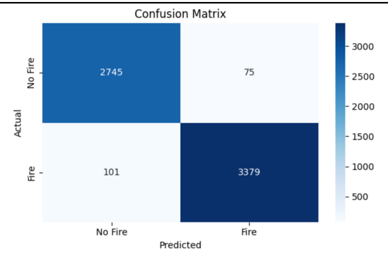
This baseline sets the foundation for future experimentation with deeper and more complex architectures.

Scaling Up: Medium and Large Models

After the baseline model, I built **medium** and **large** CNNs to improve performance and capture more complex patterns in the data.

Larger models have greater capacity to learn detailed features—important for detecting subtle wildfire indicators like smoke and burn areas. This step also helped me compare architectures, understand how model size affects accuracy, and find the right balance between complexity and generalization.

Model Comparison Table

Features	Small	Medium	Large
Number of parameters	63617	762773	3304769
Convolutional Layers	1	1	3
Filter Sizes	3x3	3x3	3x3
Dense Layers	1	2	2
Activation Functions	ReLu, Sigmoid	ReLu, Sigmoid	ReLu, Sigmoid
Training			
Validation Accuracy	93%	94%	96%
Confusion Matrix			
Best Use	Prototyping, debugging, Real time video detection	Balanced training and validation for testing	Final model for satelite detection

GUI with Gradio

The GUI is built with Gradio and consists of a clean, minimal layout. It includes an image upload component that accepts standard formats (`.jpg`, `.jpeg`, `.png`) and a dropdown menu for selecting one of three available deep learning models. Once both the image and model are provided, a "Predict" button detects wildfres. The prediction result is displayed as text below, indicating whether a wildfire was detected. The interface is responsive and runs in any modern web browser.