WILDFIRE PREDICTION ANALYSIS (Report)

Report Contents

1. INTRODUCTION

Wildfires impacts ecosystems, property, and human life worldwide. Detecting wildfires quickly from satellite or drone imagery is crucial for early response and damage mitigation. Manual monitoring is often slow, while sensor-based systems can be expensive and have limited coverage. This project builds an image-based wildfire detection system using two deep learning approaches: a custom Convolutional Neural Network (CNN) and a pretrained ResNet50 model. We use the Wildfire Prediction Dataset (Kaggle), which contains labeled images in train, valid, and test directories across two classes: wildfire and no wildfire. The main objective is to compare these models on accuracy, training time, and suitability for real-world use.

2. LITERATURE REVIEW

- **Zhang et al. (2022)**

  This study used CNNs to detect wildfires from satellite images. The model showed strong performance after applying normalization and data augmentation techniques. They achieved high accuracy across multiple test regions.

- **Yandouzi et al. (2023)**

  The authors compared CNN models like ResNet50 and VGG16 for forest fire detection. ResNet50 gave the best results, especially when combined with image augmentation. The study confirmed the usefulness of transfer learning in remote sensing.

3. METHODOLOGY

3.1 Data Cleaning and Preprocessing

The data cleaning and preprocessing steps were crucial to ensure the quality and effectiveness of the models developed for wildfire prediction. The following steps were undertaken:

Corrupted Image Removal: A function was implemented to traverse through the dataset directories and identify any corrupted images. This was done by attempting to open each image file and verifying its integrity. If an image was found to be corrupted, it was removed from the dataset. This step ensured that only valid images were used for training and evaluation, which is essential for model performance.

Image Rescaling: All images were rescaled to a uniform size of 128x128 pixels. This standardization is important as it ensures that the input to the models is consistent, allowing for efficient training.

Data Augmentation: To enhance the diversity of the training dataset and improve the model's ability to generalize, data augmentation techniques were applied. These included:

> **Rotation:** Randomly rotating images within a specified range (e.g., 20 degrees) to simulate different orientations.
>
> **Zooming:** Applying random zooms to images to help the model learn to recognize features at various scales.

Horizontal Flipping: Randomly flipping images horizontally to increase variability in the training data.

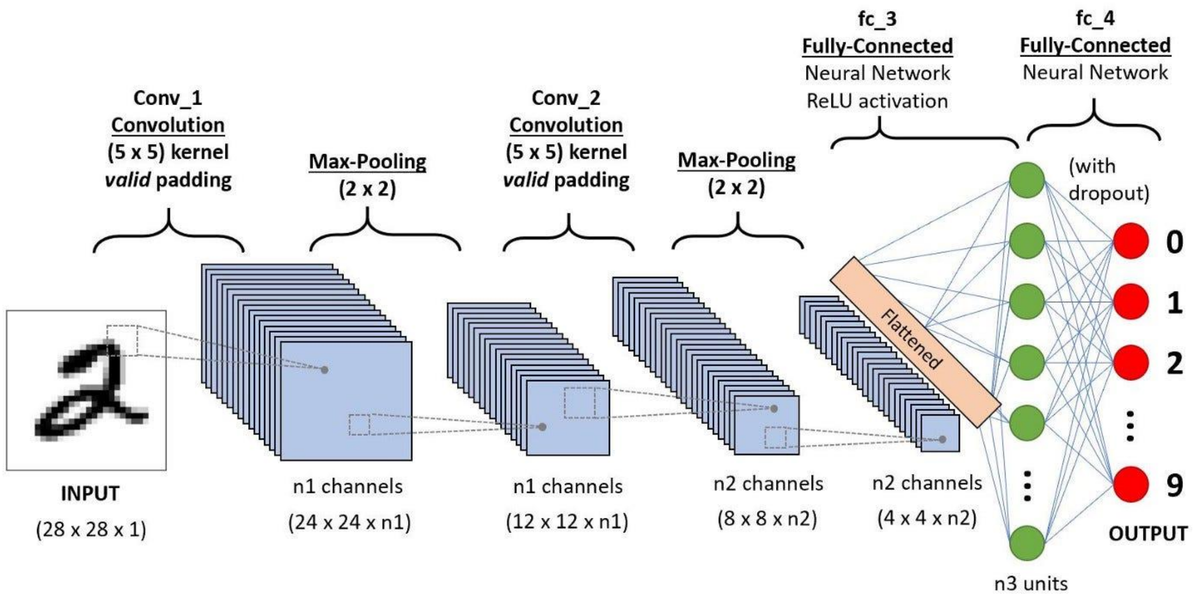
These preprocessing steps collectively improved the robustness of the models and helped mitigate overfitting.

3.2 Model Architecture and Training Details

Two different models were developed for the wildfire prediction task: a custom Convolutional Neural Network (CNN) and a pretrained ResNet50 model.

1. CNN Model

Architecture: The custom CNN was designed with the following layers:



Input Layer: Accepts images of size 128x128 pixels with three color channels (RGB).

Convolutional Layers:

- The first layer has 32 filters of size 3x3 with ReLU activation.
- The second layer has 64 filters of size 3x3 with ReLU activation.
- The third layer has 128 filters of size 3x3 with ReLU activation.

Max-Pooling Layers: After each convolutional layer, a max-pooling layer with a pool size of 2x2 is applied to reduce the spatial dimensions of the feature maps.

Flatten Layer: This layer flattens the pooled feature maps into a single vector.

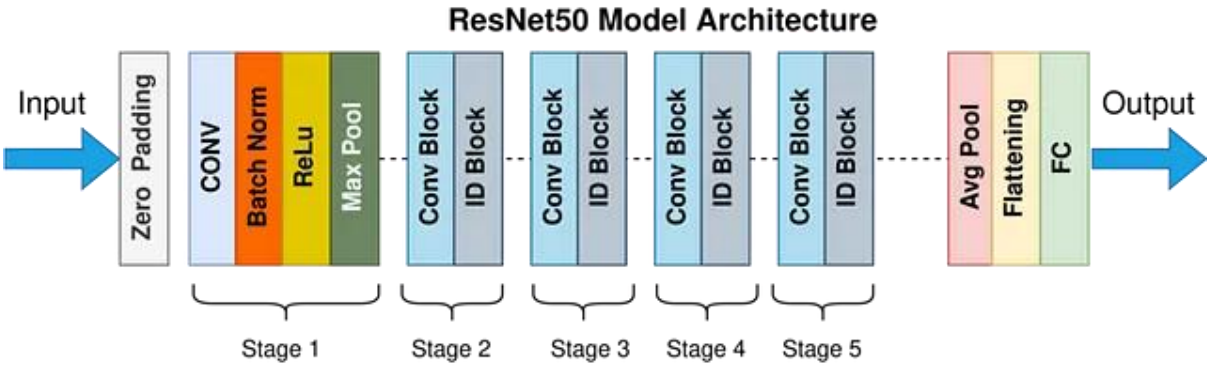
Dense Layers:

- A fully connected layer with 128 neurons and ReLU activation.
- A dropout layer with a rate of 0.2 to prevent overfitting.
- The output layer has a single neuron with a sigmoid activation function for binary classification.

Training Details: The model was compiled using the Adam optimizer with a learning rate of 0.0001. The loss function used was binary cross-entropy, which is suitable for binary classification tasks. The model was trained for three epochs on the training dataset, with validation performed on a separate validation set.

2. Pretrained ResNet50 Model

Architecture: The ResNet50 model is a deep residual network that includes:



A series of convolutional layers with skip connections, allowing gradients to flow through the network more effectively during training.

The model was modified by removing the top layers and adding a custom classifier head:

- A global average pooling layer to reduce the feature maps to a single vector.
- A dense layer with 128 neurons and ReLU activation.
- A dropout layer with a rate of 0.2.

An output layer with a single neuron and sigmoid activation for binary classification.

Training Details: Similar to the custom CNN, the ResNet50 model was compiled with the Adam optimizer and binary cross-entropy loss. The model was also trained for three epochs, leveraging the pretrained weights from ImageNet to enhance feature extraction capabilities.

3.3 Training Curves

Training curves were generated for both models to visualize their performance over the training epochs. The curves plotted include:

Accuracy Curve: This curve shows the training and validation accuracy for each epoch, indicating how well the models learned from the data.

Loss Curve: This curve illustrates the training and validation loss, providing insights into how the models minimized the loss function during training.

These curves are essential for diagnosing model performance, identifying overfitting, and understanding the learning dynamics of each model.

4. RESULTS

4.1 Training & Validation Accuracy

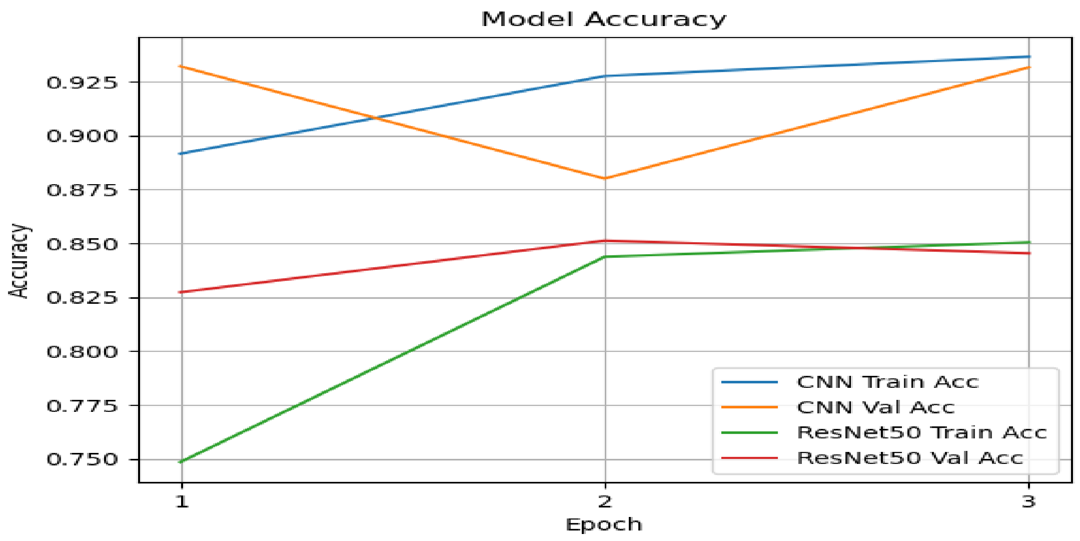


Figure shows the model accuracy over three epochs. The custom CNN model showed consistent improvement in both training and validation accuracy, eventually reaching above 94%. In contrast, the ResNet50 model plateaued earlier, with validation accuracy hovering around 86%.

4.2 Training & Validation Loss

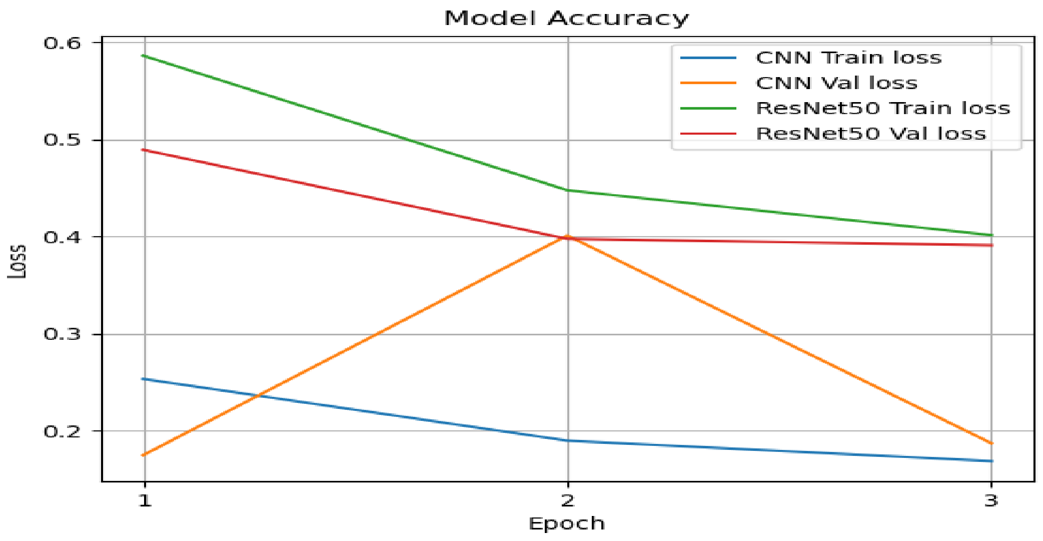


Figure highlights the loss curves for both models. The CNN model maintained a steadily decreasing loss, showing good convergence and minimal overfitting. ResNet50 also showed a decreasing trend, but its validation loss was comparatively higher, indicating less efficient learning.

4.3 Confusion Matrix

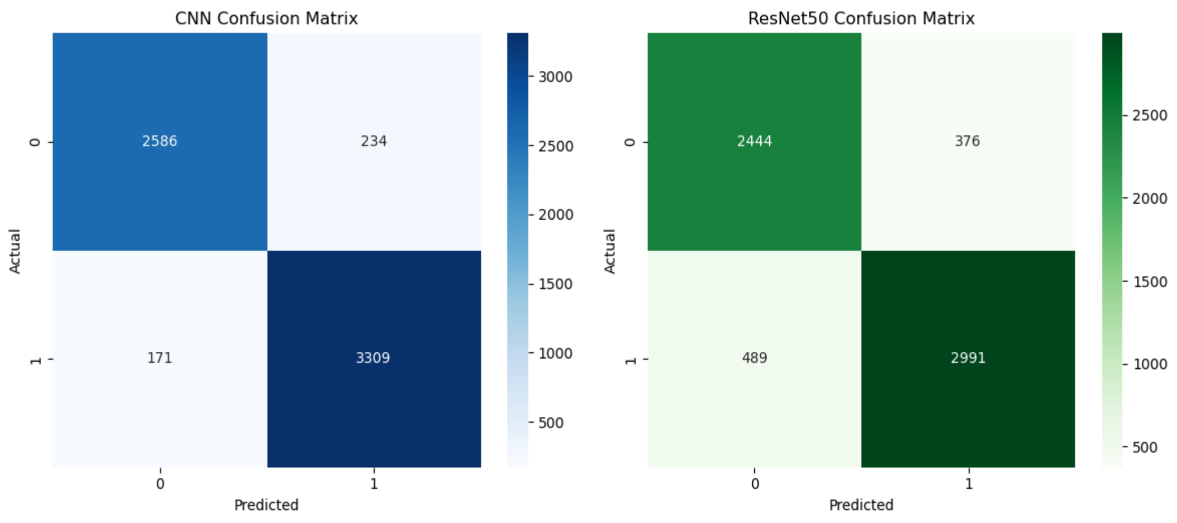


Figure compares confusion matrices for both models. The CNN model misclassified fewer images, especially for the “wildfire” class, demonstrating higher reliability. The ResNet50 model had more false negatives, which is critical in wildfire detection since failing to identify a fire can be riskier than a false alert.

4.4 ROC Curve and AUC Score

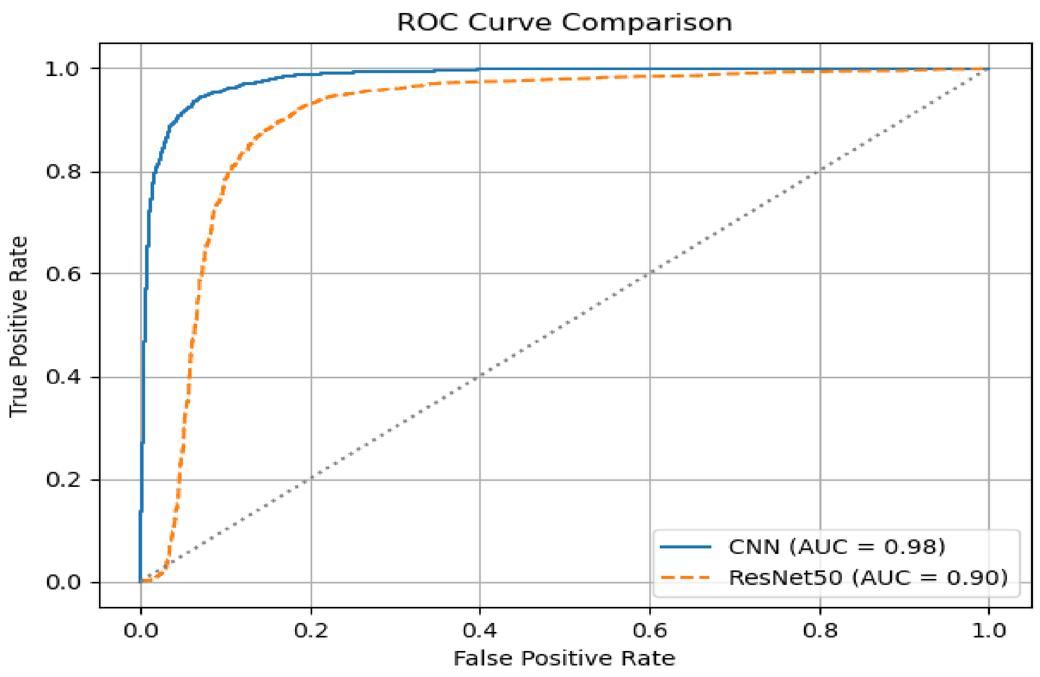


Figure shows the ROC curves. The custom CNN achieved a higher AUC score (0.98), indicating a stronger ability to distinguish between the two classes. ResNet50 had a decent curve with an AUC of 0.90, but it clearly underperformed compared to the CNN.

5. CONCLUSION

This study successfully demonstrated the application of deep learning techniques for wildfire prediction using image data. The custom Convolutional Neural Network (CNN) outperformed the pretrained ResNet50 model, achieving an accuracy of 94% compared to 86%. Key takeaways include the importance of tailored model architectures for specific tasks and the potential for improved predictive capabilities in wildfire detection. Despite the promising results, limitations such as the small dataset size and the risk of overfitting were identified. Future work should focus on expanding the dataset, increasing training epochs, and implementing regularization techniques. The practical implications of this research highlight the potential for enhanced early detection and response strategies in wildfire management.

6. REFERENCE

1. Zhang, Y., et al. (2022). Wildfire Detection Using Convolutional Neural Networks and Satellite Images. *Remote Sensing, 15*(19), 4855. https://doi.org/10.3390/rs15194855

2. Yandouzi, M., et al. (2023). Deep Learning Approaches for Wildland Fires Remote Sensing. *Remote Sensing*, *15*(7), 1821. https://doi.org/10.3390/rs15071821

3. Kaggle. (2025). *Wildfire Prediction Dataset*. https://www.kaggle.com/datasets/abdelghaniaaba/wildfire-prediction-dataset

4. TensorFlow. (n.d.). *Keras API Documentation*. https://www.tensorflow.org/api_docs/python/tf/keras