**Description of Design Choices**

The design of this model for fire, smoke, and non-fire classification involved several key decisions, which focused on optimizing both the model’s performance and its ability to handle real-world applications. Below are the main design choices made throughout the project.

**a. Choice of Deep Learning Architecture**

* **Convolutional Neural Network (CNN):** The primary design choice for the model architecture was the use of a **Convolutional Neural Network (CNN)**. CNNs are highly effective for image classification tasks because they can automatically learn spatial hierarchies in the image, identifying complex patterns and features (edges, textures, and shapes) that are essential for distinguishing between fire, smoke, and non-fire.
  + **Initial Architecture:** The custom CNN architecture was chosen to start with relatively simple layers for efficient training:
    - **Conv2D Layer**: Multiple convolution layers (with 96 and 256 filters) were applied to detect features at different levels of complexity.
    - **MaxPooling**: These layers help reduce the dimensionality of the feature maps, ensuring that only the most relevant features are passed to the next layer, preventing overfitting.
    - **Fully Connected Layers**: After flattening the feature maps, dense layers were used to interpret the learned features and output the final classification.
    - **Dropout**: Dropout layers were incorporated to prevent overfitting by randomly setting a fraction of input units to zero during training.

**b. Transfer Learning with ResNet50**

* **Pre-trained Models:** Transfer learning is an effective strategy to improve model performance, especially when the dataset is relatively small or the model training time is limited. For this task, **ResNet50**, a pre-trained model on the ImageNet dataset, was fine-tuned.
  + **ResNet50** has residual connections that allow deeper networks to train effectively without suffering from the vanishing gradient problem. This architecture is known to perform exceptionally well in various image classification tasks, and leveraging its pre-trained weights allowed the model to learn rich feature representations even with fewer data.
* **Fine-tuning Approach:** The pre-trained ResNet50 model was used with its weights frozen initially, and then the final layers were replaced with a custom fully connected layer. The top layer was fine-tuned for the fire, smoke, and non-fire classes, which improved the model’s ability to generalize to the new task.

**c. Data Augmentation**

* **Augmentation Techniques:** Given the potential for overfitting with smaller datasets, **data augmentation** was crucial for improving the model's generalization ability. Augmentation techniques such as:
  + **Random rotations, shifts, and zooms** were used to simulate variations in image perspective.
  + **Horizontal flips** ensured that the model would be invariant to the orientation of the images.

These techniques increased the effective size of the dataset and helped the model learn more robust features, thereby improving performance on unseen data.

**d. Hyperparameter Tuning**

* **Hyperparameter Selection:** Hyperparameter tuning was performed to optimize the model's performance. The following key hyperparameters were tuned:
  + **Number of filters in convolution layers**: A higher number of filters were tested for better feature extraction.
  + **Dropout rate**: A dropout rate of 0.5 was found to be effective at preventing overfitting while training.
  + **Learning rate**: The Adam optimizer with a learning rate of 0.0001 was selected to ensure gradual convergence.
  + **Batch size**: A batch size of 16 was used for training the custom CNN, while a larger batch size of 64 was used for ResNet50 due to its larger size and computational requirements.

**e. Output Layer Design**

* **Softmax Activation**: Since this is a multi-class classification task (Fire, Smoke, Non-fire), the final output layer used a **softmax activation function** to predict probabilities for each class. The softmax function ensures that the output values are between 0 and 1, and their sum equals 1, making them suitable for classification.

**Performance Evaluation of the Model**

Performance evaluation is critical to understanding how well the trained model can generalize to unseen data. Several metrics were used to evaluate both the custom CNN model and the ResNet50-based transfer learning model.

**a. Metrics Used**

* **Accuracy**: The percentage of correct predictions made by the model out of all predictions. This is the most common metric for classification tasks.
* **Loss Function**: **Categorical Cross-Entropy** loss was used to measure the difference between the true labels and the predicted probabilities. The lower the loss, the better the model’s predictions align with the true labels.
* **Confusion Matrix**: A confusion matrix was used to visualize the performance of the model across the three classes (Fire, Smoke, Non-fire). It shows the number of correct and incorrect classifications for each class, helping identify where the model is making mistakes.
* **Precision, Recall, and F1-Score**: These metrics were calculated for each class to evaluate the model’s ability to correctly identify each category and handle class imbalances.

**b. Training Results**

* **Custom CNN Model**:
  + The custom CNN model was trained for 20 epochs with early stopping implemented to prevent overfitting. The training accuracy reached over 90%, and the validation accuracy stabilized at approximately 85%.
  + **Training Loss**: The training loss decreased steadily, indicating that the model was learning and improving over time.
* **ResNet50 Model**:
  + The **ResNet50** model performed better in terms of accuracy, reaching a validation accuracy of 93%. This improvement was due to the model's ability to leverage pre-trained features, which allowed it to learn complex representations even with a smaller dataset.
  + **Training Loss**: The loss also decreased rapidly during the initial epochs, followed by a more gradual reduction as the model began fine-tuning.

**c. Comparison of Model Performance**

* The **ResNet50** model significantly outperformed the custom CNN model, especially in terms of accuracy and generalization to unseen data. This was expected due to the deep architecture of ResNet50 and its use of residual connections, which allow it to learn more complex patterns without overfitting.
* **Confusion Matrix Analysis**:
  + The custom CNN model had more false positives and false negatives compared to ResNet50, particularly for the smoke class, where images could be difficult to distinguish.
  + The ResNet50 model showed a better distribution of predictions across all three classes, with fewer misclassifications, especially between fire and non-fire categories.

**d. Real-Time Prediction**

* **Inference Time**:
  + The **custom CNN model** had a slightly faster inference time compared to ResNet50 due to its simpler architecture. However, for real-time applications, inference time was still acceptable for both models.
  + The **ResNet50 model**, while slower in inference, was suitable for applications where high accuracy is more important than real-time speed, or where edge computing devices with sufficient processing power are available.

**e. Overfitting and Generalization**

* **Overfitting**: Both models showed a slight overfitting trend when training for more than 20 epochs. The validation accuracy began to plateau, and the training accuracy continued to rise. To mitigate overfitting, **early stopping** was employed, which stopped the training process once the validation accuracy no longer improved.
* **Generalization**: The use of **data augmentation**, **dropout layers**, and **transfer learning** helped the models generalize better, even when tested on unseen data or images taken from different sources.

**Conclusion on Performance**

Both the **custom CNN model** and the **ResNet50 model** demonstrated strong performance in classifying fire, smoke, and non-fire images. The choice of architecture, data augmentation, and transfer learning played crucial roles in achieving high accuracy and ensuring the model’s robustness.

* **Custom CNN**: A solid baseline model, suitable for smaller-scale applications where real-time performance is more important.
* **ResNet50**: A more accurate and powerful model, ideal for environments where accuracy is prioritized over speed or where there is access to powerful computational resources for inference.

**Future Work and Enhancements**

* **Optimization**: Further optimizations such as quantization or pruning could be explored for improving inference speed in the ResNet50 model without sacrificing accuracy.
* **Deployment on Edge Devices**: The models could be deployed on mobile or embedded systems using frameworks like TensorFlow Lite, which would allow real-time fire and smoke detection in remote locations.