**Convolutional Neural Network (CNN) Log**

A convolutional neural network (CNN, or ConvNet) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other.

**Dataset and Preprocessing**

The dataset consists of labelled images for classification. Preprocessing involved:

* **Resizing:** Standardized input images to 128x128 pixels.
* **Normalization:** Pixel values scaled to [0,1] for stable training.
* **Data Splitting:** Dataset divided into training (80%), validation (10%), and test (10%) sets.

**CNN Architecture:**

1. **Baseline CNN Model** A simple CNN model with essential layers for feature extraction and classification.

Architecture:

* Input: 128x128 RGB images
* Convolutional Layers: 2 Conv layers (32, 64 filters) with ReLU activation
* Pooling Layers: MaxPooling (2x2) after each Conv layer
* Flatten Layer: Converts feature maps into a 1D array
* Dense Layers: Fully connected layers (128 neurons) with ReLU
* Output Layer: Softmax activation for multi-class classification

Training Performance:

* Epochs: 10
* Batch Size: 32
* Final Validation Accuracy: ~85.3%

1. **Optimized CNN Model** Using hyperparameter tuning, the CNN model was improved by optimizing:

* Filter Sizes: (3x3, 5x5)
* Number of Filters: 32, 64, 128
* Dropout Rate: 0.2-0.5
* Optimizer: Adam, RMSprop

Best Hyperparameters Selected:

* Filters: 64, 128
* Dropout: 0.3
* Learning Rate: 0.001

Final Training Performance:

* Epochs: 10
* Batch Size: 32
* Final Validation Accuracy: ~88.2%
* Test Accuracy: ~87.6%

**Evaluation and Comparison:**

The performance of the baseline and optimized CNN models was compared using accuracy and loss trends.

|  |  |
| --- | --- |
| **Model** | **Test Accuracy** |
| Baseline CNN | 85.3% |
| Optimized CNN | 87.6% |

Validation accuracy trends showed a steady improvement, with the optimized CNN outperforming the baseline.

**Conclusion:**

This study demonstrated how hyperparameter tuning enhances CNN performance in image classification tasks. The optimized CNN achieved better accuracy by adjusting filters, dropout, and optimizer settings.

**Future Enhancements:**

* Training for more epochs with a learning rate scheduler.
* Data augmentation techniques to improve model generalization.
* Experimenting with deeper CNN architectures like ResNet for better feature extraction.

This report provides a structured approach to CNN-based image classification and forms the basis for further improvements in computer vision tasks.

**Findings and Work Log Summary**

|  |  |  |  |
| --- | --- | --- | --- |
| **Phase** | **Work Done** | **Duration** | **Difficulty Level (1-10)** |
| Loading Dataset | Load image dataset, preprocess (resize, normalization, split) | 10 mins | 4 |
| CNN Model Implementation | Define CNN architecture with Conv, Pooling, and Dense layers | 50 mins | 6 |
| Model Compilation | Compile the model with optimizer, loss, and metrics | 10 mins | 3 |
| Model Training | Train the CNN on the dataset for multiple epochs | 55 mins | 7 |
| Model Evaluation | Evaluate the model using accuracy and loss on the test set | 10 mins | 5 |
| Results Comparison | Compare CNN performance with baseline models | 7 mins | 4 |