Assignment: Image Classification using CNN

Mentors: Ankit Maurya and Sachin Awasthi

December 31, 2024

Objective

The objective of this assignment is to build a Convolutional Neural Network (CNN) for classifying images using the Animals dataset. You will gain experience in image preprocessing, model building, and evaluation in image classification tasks.

Important Note for Mentee

In your previous assignment, you built a basic neural network to understand fundamental concepts like layers, neurons, and activation functions. This assignment builds on that by introducing CNNs, a more complex version of neural networks designed for image data. CNNs use convolution and pooling layers to capture spatial features, making them highly effective for image classification.

While understanding neural networks is crucial, note that CNN architectures are pre-built and optimized for various tasks. In this assignment, you will use existing models and libraries (like Keras or TensorFlow) to implement a CNN, focusing on applying these tools to real-world problems rather than building the model from scratch.

Dataset

1. Data Preparation: - Translate folder names as follows:

   - **cane** → Dog
   - **cavallo** → Horse
   - **elefante** → Elephant

- **farfalla** → Butterfly
- **gallina** → Chicken
- **gatto** → Cat
- **mucca** → Cow
- **pecora** → Sheep
- **scoiattolo** → Squirrel
- **ragno** → Spider

- Organize the dataset into:

- **Train (70%)**
- **Validation (20%)**
- **Test (10%)**

Instructions

1. Preprocessing

- Resize images to **128x128**. - Normalize pixel values to **[0, 1]**. - Apply data augmentation techniques (e.g., rotation, flipping, zoom).

2. Model Architecture

Design a CNN with the following structure:

- **Input**: 128x128x3 images.
- **Convolutional Layers**: Experiment with filters (e.g., 32, 64, 128).
- **MaxPooling Layers**: Reduce spatial dimensions.
- **Dropout Layers**: To prevent overfitting.
- **Dense Layers**: Use softmax for the output layer with 10 neurons.

3. Training

- Use the following configurations:

  - Optimizer: **Adam**.
  - Loss Function: **Categorical Crossentropy**.
  - Metrics: **Accuracy**.

- Train for **15 epochs** with training and validation sets.

4. Evaluation

Evaluate your model using the **test set** and calculate the following metrics:

- Accuracy.
- Precision, Recall, and F1-Score (using `sklearn`'s `classification_report`).

5. Visualization

- Plot **Training and Validation Accuracy** over epochs. - Plot **Training and Validation Loss** over epochs.

6. Predictions

- Load a few test images. - Display the images with predicted labels and confidence scores.

Deliverables

Submit the following:

1. Python script or Jupyter Notebook with:
   - .ipynb file of code
   - CNN model implementation.
   - Training and evaluation steps.
2. Plots for accuracy and loss over epochs.
3. Test results (accuracy and classification report).