EXPERIMENT 3

EXPERIMENT OBJECTIVE

To implement Convolutional Neural Networks (CNNs) for Image Classification on CIFAR-10 Dataset and Cat vs Dog Dataset, Using a Custom-Built ResNet-10 Architecture.

The experiment explores different CNN configurations by varying:

- **Activation Functions:** ReLU, Tanh, Leaky ReLU
- **Weight Initialization Techniques:** Xavier Initialization, Kaiming Initialization, Random Initialization
- **Optimizers:** SGD, Adam, RMSprop

Additionally, the best CNN model for each dataset is compared with a pretrained **ResNet-18** model using transfer learning.

DATA PREPROCESSING

Loading the Datasets

CIFAR-10

- **Source:** The CIFAR-10 images are organized into separate folders for each of the 10 classes.
- **Image Properties:** Each image is originally a 32×32 RGB image.
- **Transformations for Training:**
    - **Resizing:** Although CIFAR-10 images are 32×32, the resizing step is maintained for consistency.
    - **Augmentation:** Random horizontal flips are applied with a probability of 0.5.
    - **Normalization:** Images are normalized using the CIFAR-10 mean ([0.4914, 0.4822, 0.4465]) and standard deviation ([0.247, 0.243, 0.261]).
- **Transformations for Testing:** Only resizing and normalization (no augmentation).
- **Splitting:** The entire training dataset is split into 80% training and 20% validation sets using PyTorch's random_split.
- **Data Loading:** The image tensors (and corresponding labels) are wrapped in TensorDataset and loaded with a batch size of 16 using DataLoaders.

Cat vs Dog

- **Source:** The dataset contains images of cats and dogs, organized in separate folders.
- **Image Properties:** Original images are resized to 224×224 to suit the input requirements.
- **Transformations for Training:**
    - **Resizing:** All images are resized to 224×224 pixels.
    - **Augmentation:** Random horizontal flips (p = 0.5) are used to diversify the training data.
    - **Normalization:** Images are normalized using ImageNet statistics (mean: [0.485, 0.456, 0.406] and std: [0.229, 0.224, 0.225]).
- **Transformations for Testing:** Only resizing and normalization are applied.
- **Splitting:** Similar to the CIFAR-10 experiment, the training set is split into 80% training and 20% validation.
- **Data Loading:** Data is loaded into DataLoaders (batch size = 16) after stacking and converting to PyTorch tensors.

Splitting the Dataset

- Both the model is trained on 80% of the Datasets training data with periodic evaluation on the validation split.

NEURAL NETWORK IMPLEMENTATION

Architecture: ResNet-10 (Without Skip Connections)

- **Input Layer:**
    - A standard convolutional layer that maps the 3-channel input into 64 feature maps.
- **Preactivation Blocks:**
    - The network employs PreActBlock modules. Each block comprises:
        - ❖ A Batch Normalization layer followed by ReLU activation.
        - ❖ A 3×3 convolution.
        - ❖ A dropout layer (with dropout probability set to 0.3).
        - ❖ A second BN-ReLU-conv-dropout sequence.

- **Network Structure:**
  - Four sequential layers are constructed:
    - **Layer 1:** 2 blocks with 64 filters (stride = 1).
    - **Layer 2:** 2 blocks with 128 filters (stride = 2 for downsampling).
    - **Layer 3:** 2 blocks with 256 filters (stride = 2).
    - **Layer 4:** 2 blocks with 512 filters (stride = 2).
- **Final Layers:**
  - An adaptive average pooling layer converts the spatial dimensions to 1×1.
  - A fully connected layer outputs class predictions (10 classes for CIFAR-10; 2 classes for Cat vs Dog).
- **Weight Initialization:**
  - Weights in both convolutional and linear layers are initialized using the Kaiming uniform method.
- **Activation Functions:**
  - ReLU is used throughout the network.

TRAINING CONFIGURATION

Training the Model (Both Experiments)

- **Loss Function:** Cross-Entropy Loss is used for multi-class classification.
- **Optimizer:** The Adam optimizer is chosen.
- **Batch Size:** A batch size of 16 is used for training, validation, and testing.
- **Epochs and Learning Rates:**
  - **CIFAR-10:**
    - Learning Rate: 0.003
    - Number of Epochs: 500
  - **Cat vs Dog:**
    - Learning Rate: 0.001
    - Number of Epochs: 500
- **Reproducibility:** A fixed seed (42) is set across Python's random module, NumPy, and PyTorch (CPU and GPU) to ensure reproducible results.

TRAINING AND VALIDATION RESULTS

CIFAR-10 Dataset

- **Training Size:** 40,000 images
- **Validation Size:** 10,000 images
- **Best Validation Accuracy:** 80.06% at Epoch 297
- **Loss & Accuracy Progression:**
  - **Epoch 1:** Loss = 2.39, Accuracy = 17.49%, Validation Accuracy = 26.41%
  - **Epoch 10:** Loss = 1.39, Accuracy = 49.84%, Validation Accuracy = 53.50%
  - **Epoch 50:** Loss = 0.52, Accuracy = 82.03%, Validation Accuracy = 76.03%
  - **Epoch 100:** Loss = 0.22, Accuracy = 92.66%, Validation Accuracy = 77.85%
  - **Epoch 200:** Loss = 0.11, Accuracy = 96.69%, Validation Accuracy = 78.00%
  - **Epoch 297:** Loss = 0.0687, Accuracy = 97.78%, Validation Accuracy = 80.06% (Best Model)

Cat vs Dog Dataset

- **Training Size:** 6,404 images
- **Validation Size:** 1,601 images
- **Best Validation Accuracy:** 89.07% at Epoch 345
- **Loss & Accuracy Progression:**
  - **Epoch 1:** Loss = 1.57, Accuracy = 49.83%, Validation Accuracy = 51.78%
  - **Epoch 50:** Loss = 0.64, Accuracy = 64.33%, Validation Accuracy = 59.84%
  - **Epoch 100:** Loss = 0.66, Accuracy = 62.57%, Validation Accuracy = 59.59%
  - **Epoch 200:** Loss = 0.32, Accuracy = 86.48%, Validation Accuracy = 81.32%
  - **Epoch 300:** Loss = 0.22, Accuracy = 90.97%, Validation Accuracy = 83.64%
  - **Epoch 314:** Loss = 0.17, Accuracy = 93.25%, Validation Accuracy = 85.38%
  - **Epoch 345:** Loss: 0.1525, Accuracy: 93.50%, Validation Accuracy: 89.07% (Best Model)

Evaluation Results

CIFAR-10

- Final Test Loss and Test Accuracy are reported as 1.0026 and 78.64%, respectively.

Cat vs Dog

- Final Test Loss and Accuracy are computed as 0.4070 and 86.85%, respectively.

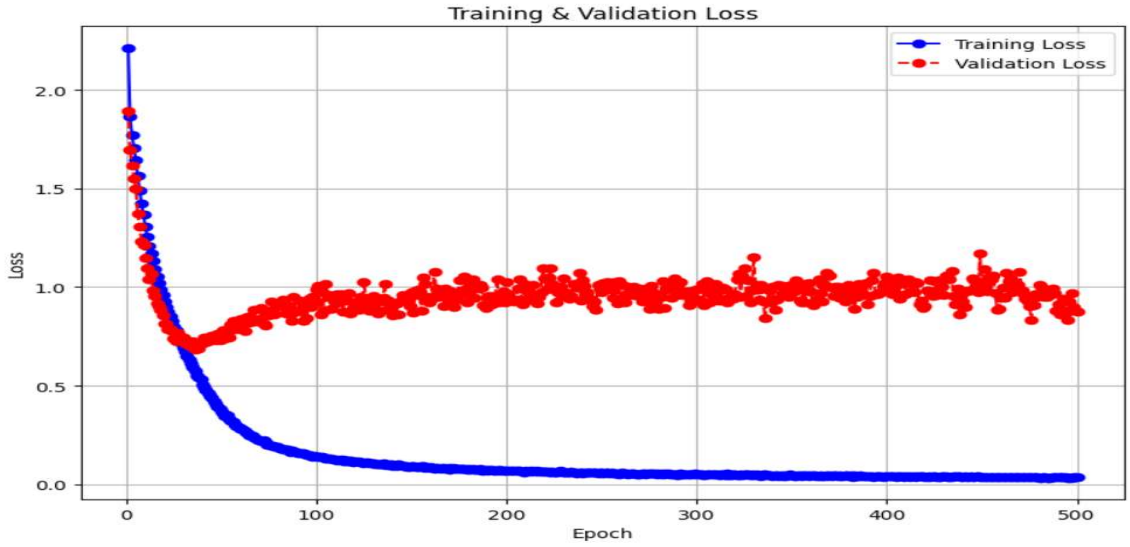
OBSERVATIONS AND CONCLUSIONS

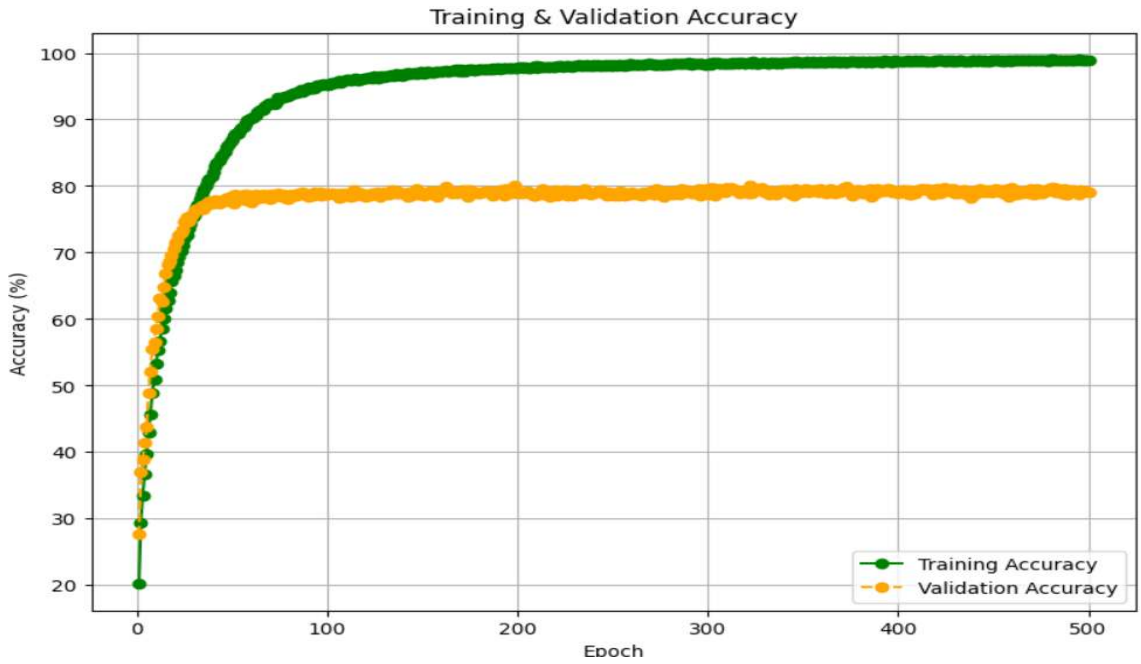
- Proper normalization and data augmentation (random horizontal flips) were crucial in achieving robust performance on both datasets.
- Even without explicit skip connections, the ResNet-10 architecture using preactivation blocks and dropout achieved competitive results.
- The use of dropout helped reduce overfitting, especially important given the relatively small batch sizes and limited training epochs.
- The CIFAR-10 dataset (with 10 classes) and the binary Cat vs Dog dataset each pose different challenges.
- The choice of input image size (32×32 vs 224×224) and normalization statistics was tailored to each dataset.
- The saved best model (based on validation performance) consistently delivered strong performance on unseen test data, highlighting good generalization.

RESULTS AND VISUALIZATION

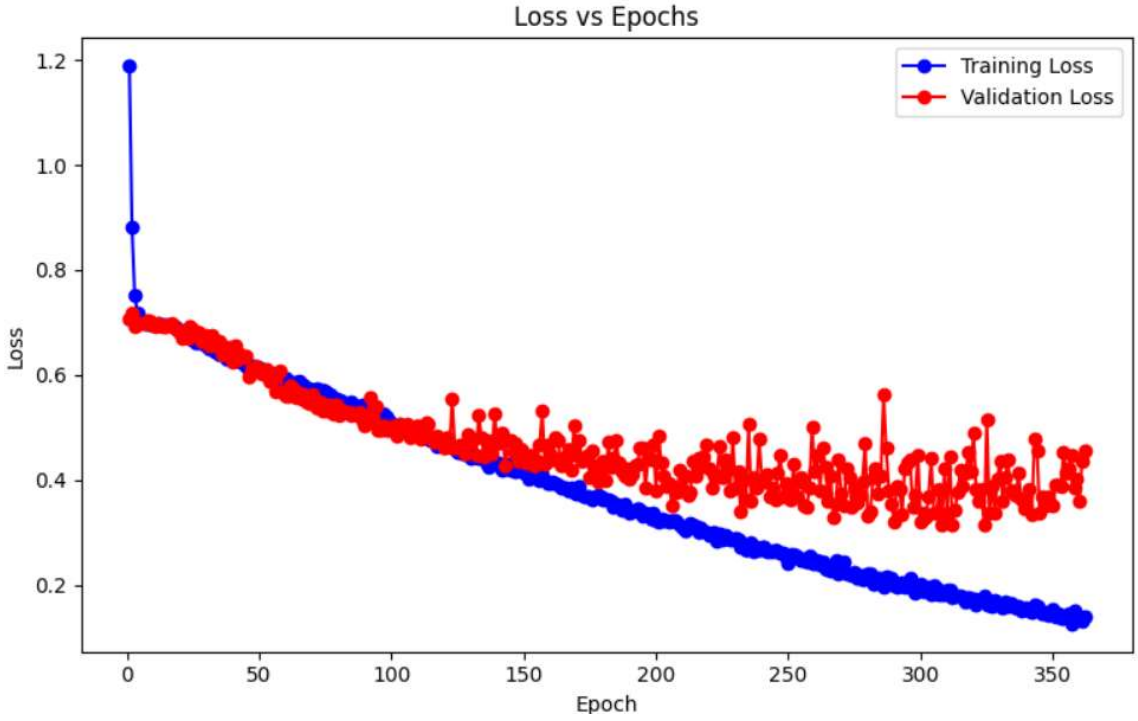
- Plots of training and validation loss as well as accuracy trends were generated for the experiments.
- Sample predictions on test images were visualized to qualitatively assess model performance.
- Results from the CIFAR-10 experiment and the Cat vs Dog experiment provide insights into how similar architectures perform on datasets with differing numbers of classes and image complexities.

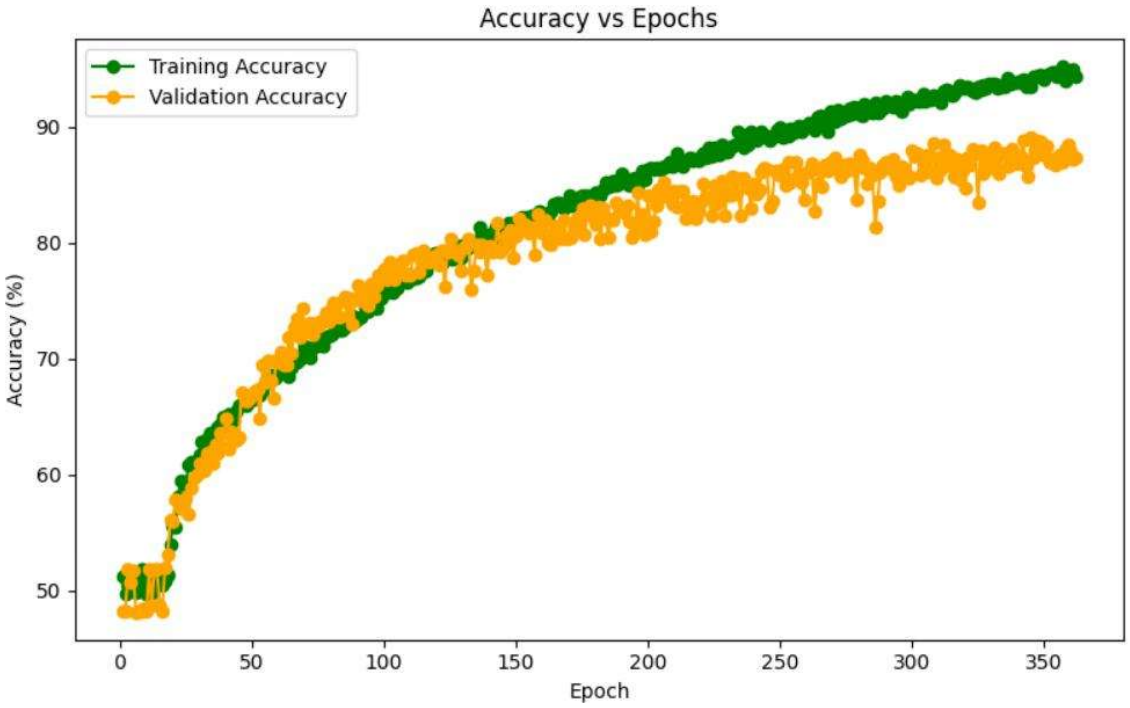
CIFAR-10




Training & Validation Accuracy
Accuracy (%)
Epoch
Training Accuracy
Validation Accuracy

CAT vs DOG


Loss vs Epochs
Loss
Epoch
Training Loss
Validation Loss



FILES INCLUDED

- Experiment_2_Classify_Linearly_Separable_Dataset.py
- Experiment_2_Classify_Linearly_Separable_Dataset.ipynb
- Documentation

LINKS

GitHub: [AyusGup/Deep-Learning-Lab: DL_LAB]

Kaggle: [https://www.kaggle.com/code/gupta712/exp-3]