Comparative Analysis of Custom CNNs vs Pretrained Image Models Using Federated Learning on CIFAR-10 Dataset

Abstract

This project explores the comparative analysis between custom Convolutional Neural Network (CNN) models and pretrained image models for image classification tasks using the CIFAR-10 dataset. The models were evaluated using federated learning, a decentralized machine learning technique where training is distributed across multiple devices without sharing raw data.

The custom CNN models were compared against well-known pretrained models such as VGG16, ResNet50, and InceptionV3, with the goal of determining the effectiveness and efficiency of each approach in the context of federated learning.

Project Overview

This project focuses on performing a comparative analysis between custom CNN architectures and pretrained models (VGG16, ResNet50, InceptionV3) on the CIFAR-10 dataset. The goal is to evaluate the performance of each model in terms of accuracy and efficiency, when trained using federated learning.

Federated learning was employed to train the models in a decentralized way across multiple clients, helping to maintain data privacy as training happens locally on client devices, with only model updates being shared.

Key Features

- **Custom CNN Models:** Three custom CNN architectures designed for image classification.
- **Pre-trained Models:** Use of VGG16, ResNet50, and InceptionV3 pretrained on ImageNet, adapted for CIFAR-10.
- **Federated Learning:** Distributed training setup that simulates a client-server architecture for decentralized learning.
- **CIFAR-10 Dataset:** A standard dataset for image classification tasks containing 60,000 32x32 color images across 10 classes.

Technologies Used

- Python
- TensorFlow
- Keras
- NumPy
- Google Colab
- Jupyter Notebook

Demo

To run a demo to understand the performance of the models:

1. Clone the project into the `Colab Notebooks` folder in your Google Drive.
2. Navigate to the `demo` folder and run the `final_demo.ipynb` file.

Note: The entire project is developed in Jupyter Notebook. Please do not try to run it locally.

Code Structure

`read_data.ipynb`
Script to load and explore the CIFAR-10 dataset for use in federated learning.

`preprocess_data_1.ipynb`
Script to preprocess the CIFAR-10 dataset for random distribution.

`preprocess_data_2.ipynb`
Script to preprocess the CIFAR-10 dataset for categorical distribution.

`custom_model_1.ipynb`
Defines the first custom CNN model.

`custom_model_2.ipynb`
Defines the second custom CNN model.

`custom_model_3.ipynb`
Defines the third custom CNN model.

`pretrained_model_1.ipynb`
Loads and retrieves embeddings using ResNet50.

`pretrained_model_2.ipynb`
Loads and retrieves embeddings using VGG16.

`pretrained_model_3.ipynb`
Loads and retrieves embeddings using InceptionV3.

`pretrained_second_layer.ipynb`

Defines the second layer of a pretrained model for further training.

`local_model_train_custom.ipynb`

Trains custom models locally using federated learning and aggregates updates.

`local_model_train_pretrained.ipynb`

Trains embeddings retrieved by pretrained models locally using federated learning and aggregates updates.

Installation Instructions

To run this project:

1. Clone the repository into your Google Drive under the `Colab Notebooks` folder:

   ```
   git clone https://github.com/your-username/your-project-name.git
   ```

2. Rename the `code` folder to `DIP_proj`.

How to Use

1. After cloning, find your folder under `Colab Notebooks`.
2. Rename the `code` folder to `DIP_proj`.
3. Run `pretrained_model_1.ipynb`, `pretrained_model_2.ipynb`, and `pretrained_model_3.ipynb` to retrieve embeddings on the dataset. The embeddings will be stored in the same folder structure.
4. Run `local_model_train_custom.ipynb` and `local_model_train_pretrained.ipynb` to perform the federated learning process.
5. Observe the metrics for each model.
6. For a demo, navigate to the `demo` folder and run the `final_demo.ipynb` file.