Assignment_2 – Image Classification

Due – Oct 30, 2024

Submission instructions in Lecture_1

Context:

Precision Agriculture and AI

Agriculture has traditionally been labor-intensive, requiring constant human intervention to monitor plant growth and manage crops. Despite technological advances, tasks like sorting and recognizing different plant species, including differentiating crops and weeds, rely heavily on manual labor. This has long-term impacts on productivity and sustainability, requiring innovative solutions to modernize the industry.

Artificial Intelligence (AI) and Deep Learning, mainly using Convolutional Neural Networks (CNNs), have the potential to revolutionize this process. AI can significantly reduce the time and labor required to identify plant species, making it possible to sort and classify seedlings more efficiently and accurately than human labor. The potential applications of this technology extend to improving crop yields, reducing the environmental impact of farming, and allowing workers to focus on more complex agricultural decisions.

Industry Overview

Caterpillar:

1. **Caterpillar's Role in Precision Agriculture:** Caterpillar has been involved in developing automated technologies for agriculture, focusing on improving efficiency and precision in field operations like spraying. Their systems use machine learning and deep learning models, including CNNs, to enhance crop targeting, minimize herbicide use, and improve spraying accuracy by detecting weeds and crops in real-time. Using CNNs in these applications allows for better decision-making in the field, reducing manual labor and chemical waste.
2. **Real-Time Recognition for UAV Sprayers Using CNN:** This research explores deep learning for UAV-based precision spraying. It demonstrates how CNNs are integrated into real-time recognition systems for UAV sprayers, significantly improving crop recognition accuracy and reducing chemical usage. UAVs equipped with CNN models can target specific crops for spraying while avoiding non-target areas, optimizing the spraying process for greater efficiency.

Read more on real-time recognition for UAV sprayers using deep learning

Objective

This project aims to use CNN to classify images of plant seedlings into their respective species. By building a classifier using the provided dataset, you will create a model capable of determining a plant's species based on an image. This has far-reaching implications in precision agriculture, where accurate plant identification is essential for crop management, weed control, and overall farm productivity.

Dataset Overview

The dataset provided contains images of unique plant species, with a total of 12 categories:

- Black grass
- Charlock
- Cleavers
- Common Chickweed
- Common Wheat
- Fat Hen
- Loose Silky bent
- Maize
- Scentless Mayweed
- Shepherd's Purse
- Small-flowered Cranesbill
- Sugar Beet

The data has been preprocessed into two files for easy access:

- images.npy: Contains the image data.
- Label.csv: Contains the labels for each image corresponding to the species.

Your task is to design and train CNN using Keras to classify these images efficiently. Due to the large dataset size, it is recommended to use a GPU runtime for faster computation.

Part 1 - Assignment Instructions- Building a Python notebook – 70%

Precision Agriculture Image Classification Project

This project involves building and enhancing Convolutional Neural Network (CNN) models to classify plant seedlings into 12 species. You will go through various stages, including exploratory data analysis (EDA), data preprocessing, model building, data augmentation, and transfer learning. Your goal is to maximize the F1 score for each model.

Sections:

1. Exploratory Data Analysis (EDA)

- Load the dataset (images.npy and Label.csv).
- Visualize the distribution of the 12 plant species classes to check for class imbalance.
- Plot a subset of images in a multi-panel grid (3 rows × 4 columns) to explore the dataset visually.
- Include sample titles for each image showing the corresponding plant species.

2. Data Preprocessing

- **Resize the images** from 128x128 to 64x64 to save compute time.
- **Encode the target variables** using one-hot encoding or label encoding as appropriate.
- **Normalize the images** by scaling the pixel values to a range between 0 and 1 to prepare the data for training.

3. Model 1: Basic CNN Model

- Build a basic Convolutional Neural Network (CNN) model using Keras.
- The architecture is up to you, but ensure the model is suitable for image classification.
- Train the model and:
  - Print out **accuracy and loss curves** during training.
  - Plot the **confusion matrix** to visualize performance across classes.
  - Generate a **classification report** (precision, recall, and F1-score) for all classes.
  - Visualize a few **test samples** with their predicted and true labels.
- Use **early stopping** if necessary to avoid overfitting.

4. Model 2: Enhanced CNN with Data Augmentation and Regularization

- Enhance the model from Model 1 by introducing **data augmentation techniques** (e.g., rotation, zoom, flips, etc.).
- Apply **regularization techniques** such as:
  - **Batch Normalization** to stabilize and speed up training.
  - **Spatial Dropout** to reduce overfitting.
- Retrain the model and:
  - Print all relevant plots (accuracy and loss curves, confusion matrix, classification report).
  - Visualize test samples with their predicted and true labels.

5. Model 3: Transfer Learning with Pre-trained Models

- Select two pre-trained models of your choice (e.g., **VGG16**, **ResNet**, etc.) and apply **transfer learning** to the dataset.
- Fine-tune the top layers and add a **DNN layer** with **regularization** (e.g., dropout) to improve generalization.
- Optionally, for **extra credit**, use **Keras Tuner** to optimize the DNN layer's hyperparameters.
- Train the model and print all relevant plots (accuracy/loss curves, confusion matrix, classification report, test sample visualizations).

6. Model Comparison and F1 Score Analysis

- Compare the **F1 scores** across all three models.
- Store the F1 scores in a **DataFrame** and create a **bar plot** to visualize the comparison.
- The goal is to **maximize the F1 score** of your models.

Notes:

- Use **GPU runtime** for faster computation, especially for Models 2 and 3.
- You can experiment with the number of epochs, layers, and other architectural details, but the final goal is maximizing the F1 score.

Part 2 - Presentation Deck: (30%)

Slide 1: Project Objective (What You Did)

- **Title:** Precision Agriculture Plant Classification Using CNNs

- **Objective:**
  - Built multiple CNN models to classify plant species.
  - Used data augmentation, regularization, and transfer learning techniques to improve performance.
  - Goal: Maximize F1 score for each model.
- **Outline of the Presentation:**
  - EDA & Preprocessing
  - Model 1: Basic CNN
  - Model 2: Enhanced CNN with Augmentation
  - Model 3: Transfer Learning
  - Comparison of Models

Slide 2: Exploratory Data Analysis (EDA)

- **Title:** Data Analysis & Class Distribution
- **Content:**
  - Visualize the distribution of the plant species classes (e.g., bar chart showing imbalance).
  - Multi-panel image plot (3x4 grid) of sample images from the dataset.
  - Insights from EDA: Mention any observed class imbalances or outliers.

Slide 3: Data Preprocessing

- **Title:** Data Preprocessing Steps
- **Content:**
  - Explain the resizing of images from 128x128 to 64x64.
  - Encoding of the target labels (one-hot encoding).
  - Normalization of image pixel values to [0, 1].
  - Brief mention of how these steps affect model training (e.g., efficiency, accuracy).

Slide 4: Model 1 - CNN Architecture

- **Title:** Model 1: Basic CNN Architecture
- **Content:**
    - Draw and label the CNN architecture (number of layers, convolutional, pooling layers, etc.).
    - Include key hyperparameters like learning rate, optimizer, and number of epochs.
    - Explain why you chose this specific architecture.

Slide 5: Model 1 - Training Results

- **Title:** Model 1: Training Performance
- **Content:**
    - Show the **accuracy and loss curves** for training and validation.
    - Display the **confusion matrix** for test set predictions.
    - Present the **classification report** (precision, recall, F1-score).
    - Insights: Summarize the performance of this basic model, including its strengths and weaknesses.

Slide 6: Model 2 - Data Augmentation & Architecture

- **Title:** Model 2: Enhanced CNN with Data Augmentation
- **Content:**
    - Explain the data augmentation techniques used (rotation, flips, zoom, etc.).
    - Draw and label the updated CNN architecture, emphasizing changes from Model 1.
    - Mention regularization techniques applied (e.g., batch normalization, spatial dropout).

Slide 7: Model 2 - Training Results

- **Title:** Model 2: Training Performance
- **Content:**
    - Show **accuracy and loss curves** after using augmentation and regularization.
    - Display the **confusion matrix** and **classification report**.

- Insights: Discuss improvements in performance compared to Model 1.

Slide 8: Model 3 - Transfer Learning Setup

- **Title:** Model 3: Transfer Learning with Pre-Trained Models
- **Content:**
  - Mention the two pre-trained models you selected (e.g., VGG16, ResNet, etc.).
  - Explain how you fine-tuned these models (e.g., freezing layers, retraining certain layers).
  - Briefly describe any additional regularization techniques (e.g., dropout, L2 regularization) in the DNN layer.

Slide 9: Model 3 - Training Results

- **Title:** Model 3: Transfer Learning Performance
- **Content:**
  - Show **accuracy and loss curves** for the transfer learning models.
  - Display the **confusion matrix** and **classification report**.
  - Insights: Highlight the performance improvements from using pre-trained models.

Slide 10: F1 Score Comparison

- **Title:** Comparison of F1 Scores Across Models
- **Content:**
  - Create a bar plot comparing the F1 scores of Model 1, Model 2, and Model 3.
  - Discuss which model performed best and why.
  - Mention any trade-offs (e.g., training time, complexity, etc.).

Slide 11: Key Learnings & Challenges

- **Title:** Key Learnings from the Project
- **Content:**

- Summarize the key takeaways (e.g., importance of data augmentation, value of transfer learning).
- Mention any challenges faced during training (e.g., overfitting, class imbalance) and how they were addressed.

Slide 12: Conclusion & Future Work

- **Title:** Conclusion and Future Improvements
- **Content:**
    - Summarize the overall model performance.
    - Suggest potential improvements for future work (e.g., different architectures, more advanced augmentation techniques).
    - Final thoughts on how this project could impact precision agriculture.

Guidelines:

- Use a clean and structured coding approach, separating preprocessing, model building, and evaluation steps.
- Ensure that you document your code well, especially the parts involving Keras Tuner and model evaluation.
- Provide clear business insights from your results and predictions.