Macquarie University

Sydney, Australia

Evaluating the Efficacy of Local vs. LLM Models in Plant Disease Prediction Across Diverse Species

Phuc Tuong Ngo

Supervised by
Dr. Qiongkai Xu, Dr. Xiaohan Yu

March 2025

Abstract

This research aims to benchmark the performance of local machine learning model against Large Language model (LLM) in regard to image classification for plant disease detection.

The accuracy of the model is critical to enhance agriculture and promote sustainability. Given the importance of this, the case study utilises a comparative analysis to evaluate both models across a variety of plant species and sub-species. By incorporating a subset of Plant Leaf Diseases Training Dataset, we have carried out extensive testing on both local and LLM, measuring the performance in terms of accuracy and precision across various benchmarks.

The findings reveal noticeable differences with local model demonstrating outstanding performance in prediction accuracy. These result suggests directions for future research in further local model optimisation, tailoring to agricultural needs. This study contributes to the application of advanced AI techniques in agricultural industry and proposal for viable research directions for practical application of these technologies in real-world settings.

Introduction

The application of machine learning, specifically in image classification has revolutionised disease detection as suggested by (1). However, the usage of such technology mobilely and accurately, especially in the agricultural environment has presented conundrums. This research paper aims to deliver a comparison in the performance of a fine tuned local model, in comparison to advanced large language models in terms of image classification for plant disease detection.

- **Research Question:** How do local machine learning models perform in comparison to Large Language Models in regards of accuracy in plant disease detection across various plant's species and sub-species.
- **Significance:** The importance of this research grows beyond the pursuit of academia, focusing to provide real-world benefit for the agricultural industry.

Methodology

0.1 Data Collection

The dataset was sourced from Kaggle's "Plant Leaf Disease Training Dataset, which is a combination of a variety of different sub-datasets such as Cassava Leaf Disease Dataset, Rice Leaf Disease Images, PlantVillage, Potato Leaf Disease. This aggregation dataset provides a diverse variety of plant species such as apple, orange, peach, potato, each with specific disease such as black rot, rust scab, bacterial spot as well as healthy specimens.

0.2 Data Selection

The whole dataset is around too large to be benchmarked by LLM. To ensure a balanced, unbiased representation, a random selection script was written to choose 70 images from each sub-category for each plant species. This approach will ensure that each plant species

and its respective sub-category are equally represented. Therefore providing a balanced subset of data for model benchamarking.

```
import os
import random
import shutil

# Path configuration
SOURCE_DIR = "path/to/full/dataset"
DEST_DIR = "path/to/subset"
os.makedirs(DEST_DIR, exist_ok=True)

# Randomly select and copy images
for category in os.listdir(SOURCE_DIR):
    images = os.listdir(os.path.join(SOURCE_DIR, category))
    selected_images = random.sample(images, 70)
    for image in selected_images:
        src = os.path.join(SOURCE_DIR, category, image)
        dst = os.path.join(DEST_DIR, category, image)
        shutil.copy(src, dst)
```

1 Model Selection

This study used two models to analyse the correctness of image classification techniques for plant disease detection: Local Machine Learning Model and Large Language Model (LLM).

1.0.1 Local Machine Learning Model: PlantDiseaseDetectorVit2

The local model selected for this study was the "PlantDiseaseDetectorVit2" from (2), fined-tuned for plant disease detection by Abhiram and hosted on Hugging Face. The base model is vit-base-patch16-224, an open-source Vision Transformer pre-trained on ImageNet-21k, in a supervised environment from (3).

The Model works by dividing images into patches, each patch is linearly embedded into a fixed-size feature vector which converts the image into a sequence of vectors. A classification token is inserted at the beginning of each sequence which serves as a meta data providing additional information about the image. Positional Embeddings are added to each patch vector as well as the token to help the model understands the spatiality of the patches relative to the original image. The sequence of the embedded patches are passed through a series of encoder layers, each with a self-attention mechanism and feed-forward networks, which allowes the model to learn patterns within the data. In essence, ViT treat the images as patches, similarly to how sentences are tokenised. The Positional Embedding strikes similarity in how words are processed in NLP.

1.0.2 Large Language Model: Gemini 2.0 Flash

Gemini 2.0 Flash, free tier was selected for its robust performance in handling complex data as well as its vision capabilities.

2 Model Benchmarking

The evaluation of the models was made using a python script that automatically runs the model and record the answer. The results were collected and saved to a CSV file for furthe analysis.

2.0.1 Local Machine Learning Model: PlantDiseaseDetectorVit2

```
def benchMarkLocal():
count = 0
results = []
for folder in os.listdir(SUBSET_DIR):
    folder_path = os.path.join(SUBSET_DIR, folder)
    for img_file in os.listdir(folder_path):
        img_path = os.path.join(folder_path, img_file)
        base64_img = encode_image_to_base64(img_path)
        predicted = predict(base64_img)
        print("finish img", count)
        count = count + 1
        results.append([folder, img_file, predicted["class"]])

csv_path = os.path.join(CURRENT_DIR, "benchmark_results.csv")
with open(csv_path, mode='w', newline='') as file:
    writer = csv.writer(file)
    writer.writerow(["True Label", "Filename", "Predicted Label"]) # CSV header
    writer.writerows(results)
```

2.0.2 Large Language Model: Gemini 2.0 Flash

```
def benchMarkGemini():
count = 0
results = []
for folder in os.listdir(SUBSET_DIR):
    folder_path = os.path.join(SUBSET_DIR, folder)
    for img_file in os.listdir(folder_path):
        img_path = os.path.join(folder_path, img_file)
        image = PIL.Image.open(img_path)
        predicted_text = "ERROR: No response"
        retry = 10
        while retry > 0:
            try:
                response = model.generate_content(["You are a plant expert, your
                    expertise is in plant's diagnosis. Your job is to diagnose this
                    plant and respond only with the disease name. If it's healthy,
                    respond only with healthy", image])
                predicted_text = response.text.strip()
                break
            except ResourceExhausted as e:
                retry -= 1
                print("Rate limited. Sleeping for 10 seconds...")
                time.sleep(10)
            except Exception as e:
                predicted_text = f"ERROR: {e}"
                break
```

```
            print(f"Finished image {count}: {img_file}")
            count += 1
            results.append([folder, img_file, predicted_text])
            if count % 500 == 0:
                print("Saving intermediate results at count =", count)
                save_results_to_csv(results, f"benchmark_gemini_results_{count}.csv")
    save_results_to_csv(results, "benchmark_gemini_results_final.csv")
```

Models for evaluation

The evaluation of local machine learning model and Large Language Model (LLM) is critical to determine the accuracy of practical applications. For this study, BERT and spaCy was used to assess the similarity in the predicted label, in comparison to the true label.

2.1 spaCy evaluation

spaCy is an open-source library for NLP in python which is utilized in this study to check the similarity in the true label compares to the prediction of the model.

```
    import spacy
# Load a pre-trained NLP model
nlp = spacy.load("en_core_web_lg") # Medium model that includes word vectors

# Convert text to doc objects
def similarity_spacy(true_label, predictions):
    true_label_doc = nlp(true_label)
    prediction_docs = nlp(predictions)
    # prediction_docs = [nlp(pred) for pred in predictions]
    similarity = true_label_doc.similarity(prediction_docs)
    return similarity
```

2.2 BERT evaluation

BERT is a transformer model from Google, (4). The model has two main versions: cased and uncased. For this study, the base model uncased was selected.

```
# Load model directly
from transformers import AutoTokenizer, AutoModel
import torch
tokenizer = AutoTokenizer.from_pretrained("google-bert/bert-base-uncased")
model = AutoModel.from_pretrained("google-bert/bert-base-uncased")

def similarity_bert(true_label, predictions):
    # Tokenize the text
    encoded_true_label = tokenizer(true_label, return_tensors="pt", padding=True,
        truncation=True)
    encoded_prediction = tokenizer(predictions, return_tensors="pt", padding=True,
        truncation=True)
    # encoded_predicted_labels = [tokenizer(label, return_tensors="pt", padding=True,
        truncation=True) for label in predictions]

    # Get embeddings without gradient calculation
```

```
    with torch.no_grad():
        true_label_embedding =
            model(**encoded_true_label).last_hidden_state.mean(dim=1)
        prediction_embedding =
            model(**encoded_prediction).last_hidden_state.mean(dim=1)

    from torch.nn.functional import cosine_similarity

    # Calculate cosine similarities
    similarities = cosine_similarity(true_label_embedding, prediction_embedding,
        dim=1).item()
    return similarities
```

Analysis

This section details the analysis of the data collected from Plant Leaf Disease Training Dataset. Every steps from intial data handling to model evaluation is discussed to ensure reproducibility and transparency in research findings.

Exploratory Data Analysis (EDA)

Exploratory Data Analysis was conducted to gain insights into the distribution and characteristics of the data.

2.2.1 Local Machine Learning Model: PlantDiseaseDetectorVit2

Local Model

```
df_local.shape
```

```
(2590, 3)
```

```
df_local.head()
```

	True Label	Filename	Predicted Label
0	Apple___alternaria_leaf_spot	112927.jpg	Tomato___Late_blight
1	Apple___alternaria_leaf_spot	112928.jpg	Apple___Cedar_apple_rust
2	Apple___alternaria_leaf_spot	112931.jpg	Apple___Black_rot
3	Apple___alternaria_leaf_spot	112934.jpg	Apple___Cedar_apple_rust
4	Apple___alternaria_leaf_spot	112935.jpg	Apple___healthy

```
df_local.describe
```

```
<bound method NDFrame.describe of                       True Label    Filename  \
0     Apple___alternaria_leaf_spot  112927.jpg
1     Apple___alternaria_leaf_spot  112928.jpg
2     Apple___alternaria_leaf_spot  112931.jpg
3     Apple___alternaria_leaf_spot  112934.jpg
4     Apple___alternaria_leaf_spot  112935.jpg
...                            ...         ...
2585        Potato___mosaic_virus  112877.jpg
2586        Potato___mosaic_virus  112879.jpg
2587        Potato___mosaic_virus  112892.jpg
2588        Potato___mosaic_virus  112904.jpg
2589        Potato___mosaic_virus  112920.jpg

                              Predicted Label
0                         Tomato___Late_blight
1                      Apple___Cedar_apple_rust
2                          Apple___Black_rot
3                      Apple___Cedar_apple_rust
4                             Apple___healthy
...                                       ...
2585                      Tomato___Late_blight
2586  Tomato___Spider_mites Two-spotted_spider_mite
2587                   Strawberry___Leaf_scorch
2588  Tomato___Spider_mites Two-spotted_spider_mite
2589                      Tomato___Late_blight

[2590 rows x 3 columns]>
```

Figure 1: Exploratory Data Analysis results showing the description of the local model.

2.2.2 Large Language Model: Gemini 2.0 Flash

Gemini

```
df_gemini.shape
```

```
(2590, 3)
```

```
df_gemini.head()
```

	True Label	Filename	Gemini Prediction
0	Apple___alternaria_leaf_spot	112927.jpg	Spider mites
1	Apple___alternaria_leaf_spot	112928.jpg	Apple rust
2	Apple___alternaria_leaf_spot	112931.jpg	Leaf spot disease
3	Apple___alternaria_leaf_spot	112934.jpg	Cedar-apple rust
4	Apple___alternaria_leaf_spot	112935.jpg	healthy

```
df_gemini.describe
```

```
<bound method NDFrame.describe of                      True Label    Filename      Gemini Prediction
0     Apple___alternaria_leaf_spot  112927.jpg          Spider mites
1     Apple___alternaria_leaf_spot  112928.jpg            Apple rust
2     Apple___alternaria_leaf_spot  112931.jpg      Leaf spot disease
3     Apple___alternaria_leaf_spot  112934.jpg       Cedar-apple rust
4     Apple___alternaria_leaf_spot  112935.jpg               healthy
...                            ...         ...                   ...
2585        Potato___mosaic_virus  112877.jpg   Potato virus Y (PVY)
2586        Potato___mosaic_virus  112879.jpg               healthy
2587        Potato___mosaic_virus  112892.jpg  Potato leafroll virus
2588        Potato___mosaic_virus  112904.jpg               healthy
2589        Potato___mosaic_virus  112920.jpg    Nutrient Deficiency

[2590 rows x 3 columns]>
```

Figure 2: Exploratory Data Analysis results showing the description of the LLM.

Data Cleaning

Data cleaning methods were used to clean the data of null, None, NA values before further analysis is conducted. There were no null, none, na value found in the dataset.

Data Cleaning ¶

```
df_local.isna().sum().sum()
```

```
0
```

```
df_local.isnull().sum().sum()
```

```
0
```

```
df_gemini.isna().sum().sum()
```

```
0
```

```
df_gemini.isnull().sum().sum()
```

```
0
```

Figure 3: Data Validation Results showing the absence of null, None, or NA values across the entire dataset.

Data Transformation

Data transformation is a crucial step in preparing the dataset for effective model evaluation. The data in the model has been re-grouped into streams that fits the purpose of the analysis

```
df_group = df_gemini.copy()
# Split the 'True Label' column and convert to lowercase
df_group[['Plants', 'Disease']] = df_group['True Label'].str.split('___',
    expand=True).map(lambda x: x.lower() if isinstance(x, str) else x)
```

```
df_group.pop('True Label')

# Insert at the beginning
df_group.insert(0, 'True Label', df_group.pop('Disease'))
df_group.insert(0, 'Plants', df_group.pop('Plants'))

# To lower case of True Label
df_group["True Label"] = df_group["True Label"].str.replace("_"," ")

# Insert Local prediction to lower case
df_group.insert(len(df_group.columns), "Local Prediction",df_local["Predicted
    Label"].str.split('___', expand=True)[1].str.replace("_"," ").map(lambda x:
    x.lower() if isinstance(x, str) else x))

#Insert Gemini Prediction to lower case
gemini_prediction = df_group.pop('Gemini Prediction')
df_group.insert(len(df_group.columns), "Gemini
    Prediction",gemini_prediction.str.replace("-"," ").map(lambda x: x.lower() if
    isinstance(x, str) else x))


filename = df_group.pop('Filename')
df_group.insert(len(df_group.columns), 'Filename', filename)
```

2.3 Analysis without Plant Names in Prediction

This section evaluates the performance of the local model and the Large Language Model if the plant names isn't explicitly provided in their answer, purely based on the disease name. This was created as not all predictions provided the label.

2.3.1 Group by Plants, True Label

spaCy analysis

As demonstrated in Figure 4, according to spaCy, the local prediction models outperform Gemini in most labels. Only a few label such as cassava healthy and coffee healthy that Gemini is more accurate in terms of prediction.
Gemini display noticeably poor performance with labels such as grape, blackrot. In contrast, in the same label, Local prediction excels in prediction.
Interestingly, both models are very accurate when it comes to predicting the disease of cherry, healthy. In comparison, the models' performance in potato, mosaic virus is mediocre.

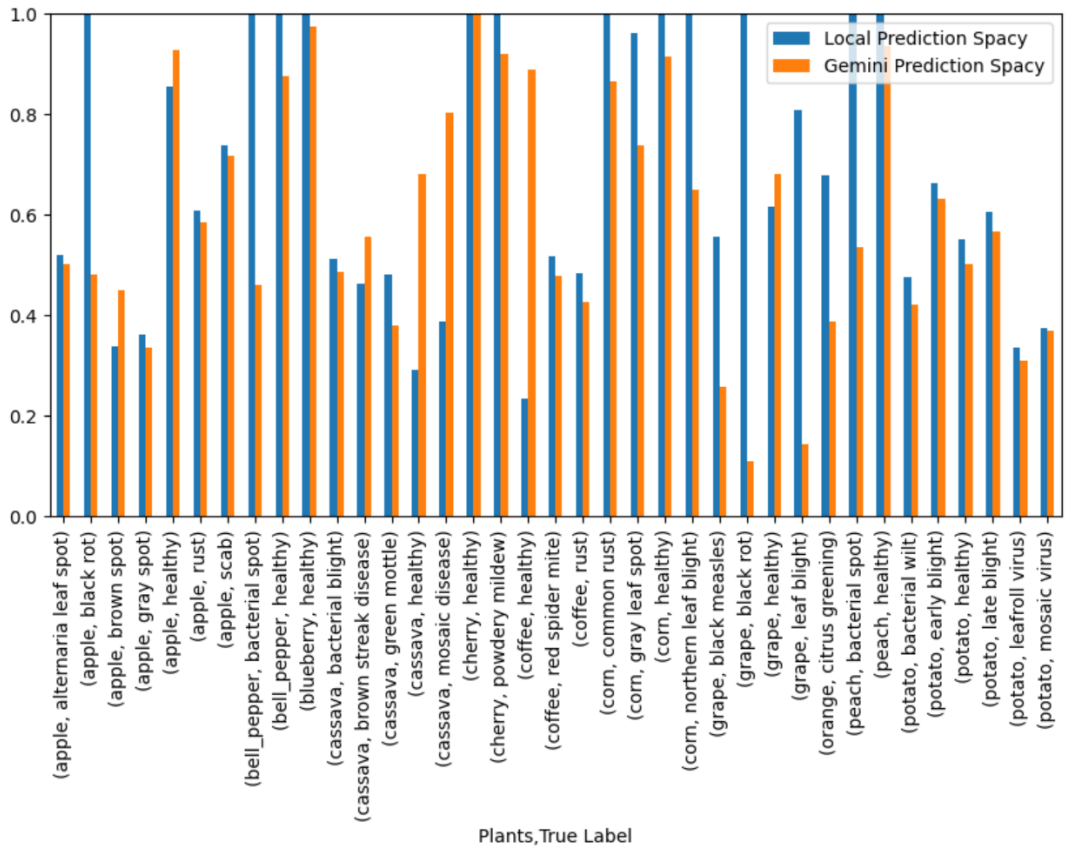

Figure 4: Comparative Accuracy of Local Prediction and Gemini Prediction Across Different Plant Diseases with spaCy, grouped-by plants, true label.

BERT analysis

Based on Figure 9, BERT's evaluation is similar to spacy in regards to the performance the models. Overall, the local model outperform gemini in most categories. The gap in performance of the two model is smaller with BERT than with spaCy.
Interstingly, in labels where the disparity between the model is largest with spaCy, BERT's evaluation is different. For instance with coffee, healthy, and grape, black rot, spaCy shows the differences to be at least one fourth, BERT displays less than a half.

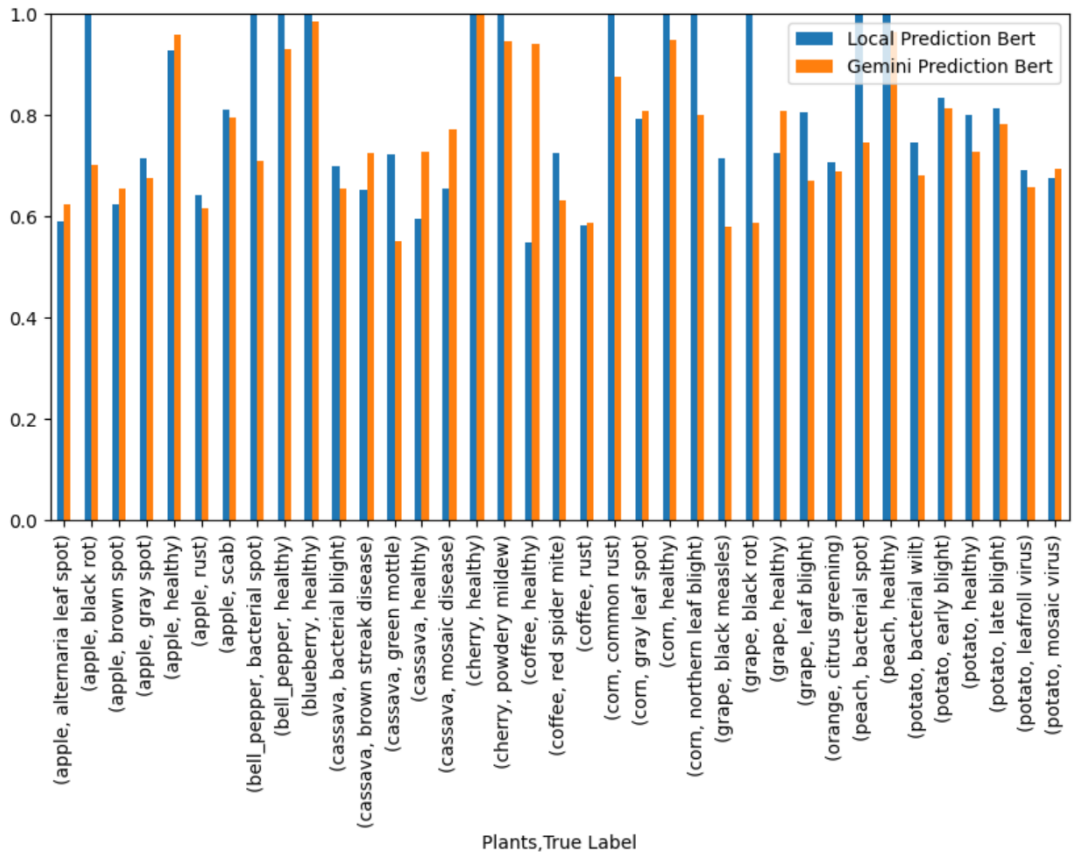

Figure 5: Comparative Accuracy of Local Prediction and Gemini Prediction Across Different Plant Diseases with BERT, grouped-by plants, true label.

2.3.2 Group by Plants

spaCy analysis

The Figure 6 displays the results consistent with the previous section where the Local Model performance surpass the Large Language Model on average with spaCy.
It is confirmed that the observation of lables such as coffee and cassava is where Gemini outperform the local model. Furthermore, grape category is where the discrepancy between the two model is the largest.
Interstingly, the local model is accurate with most categories except for cassava, coffee and potato where similarity to true label is less than 0.5
The highest image category where both model excels is blueberry and cherry on average.

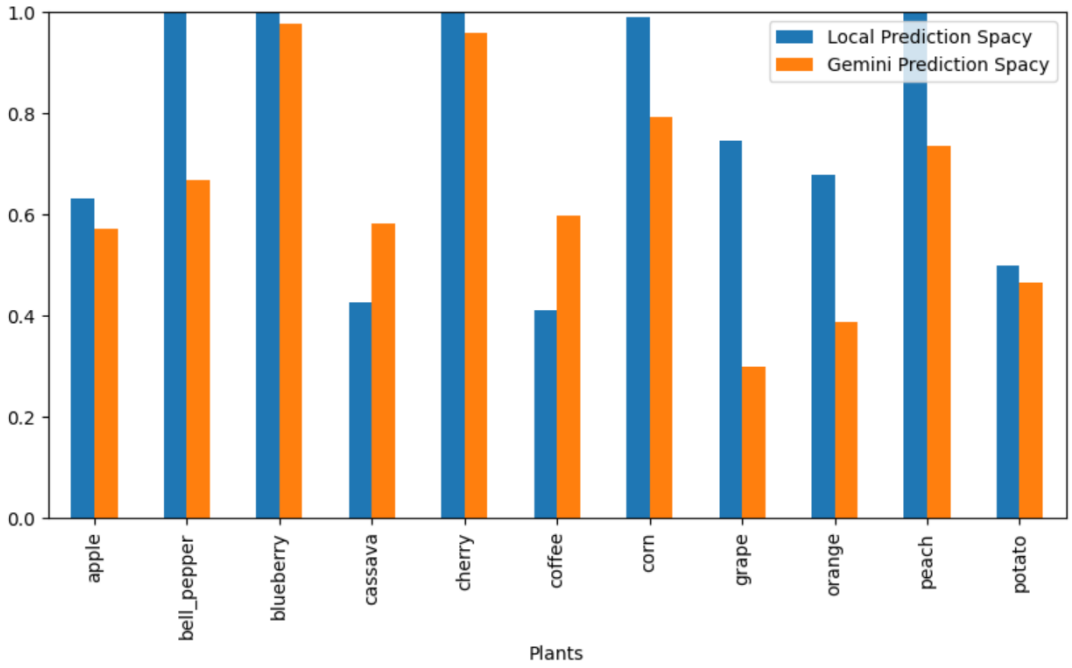

Figure 6: Comparative Accuracy of Local Prediction and Gemini Prediction Across Different Plant Diseases with spaCy, grouped-by plants.

BERT analysis

BERT's Figure 7 consolidates that Gemini is outperformed by the local prediction model with most category achieving perfect accuracy (1.0). Furthermore, Gemini matches the local model in sections such as blueberry, cherry, corn, and peach.
BERT's evalution suggests that no model's similarity is less than 0.6, in contrast to spacy's evalution.

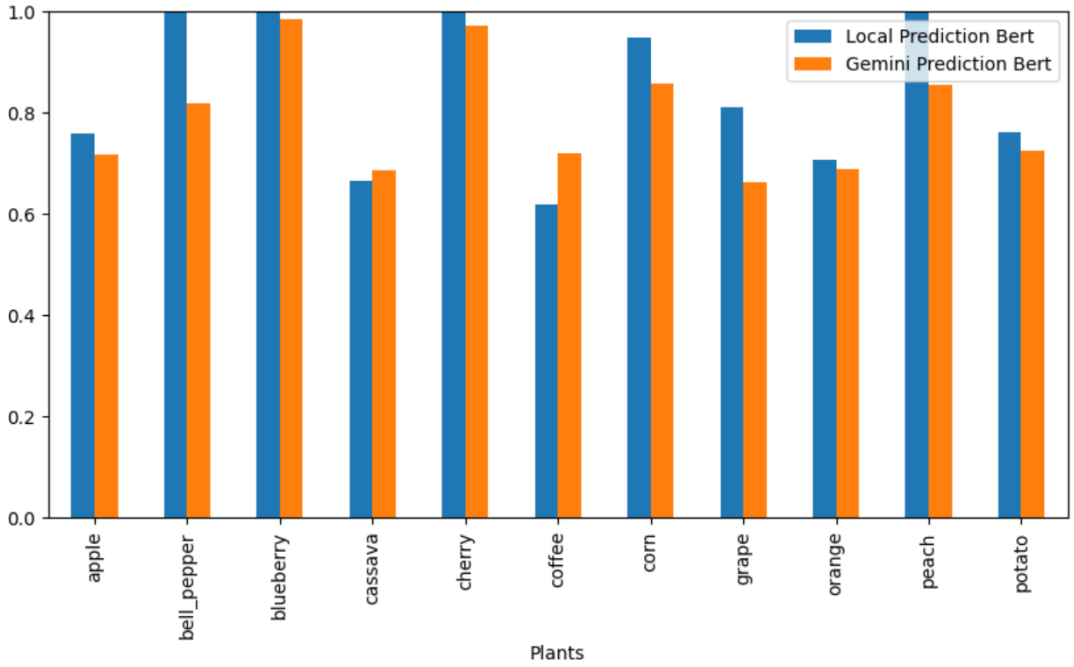

Figure 7: Comparative Accuracy of Local Prediction and Gemini Prediction Across Different Plant Diseases with BERT, grouped-by plants.

2.4 Analysis with Plant Names in Prediction

This section evaluates the performance of the local model and the Large Language Model if the plant names are provided in their answer, purely based on the disease name. By incorporating the name inside the prediction for evaluation, the prediction can be evaluated more thoroughly.

2.4.1 Group by Plants with true labels

spaCy analysis

Overall, the result is from spaCy consistent with 2.3 where the local model outperforms Gemini.
Interstingly, noth model's performance unchanges with grape. Gemini's results fluctates more across the categories in comparison to the analysis of 2.3.1. The local model's performance slightly decresse in bell pepper, corn sections.

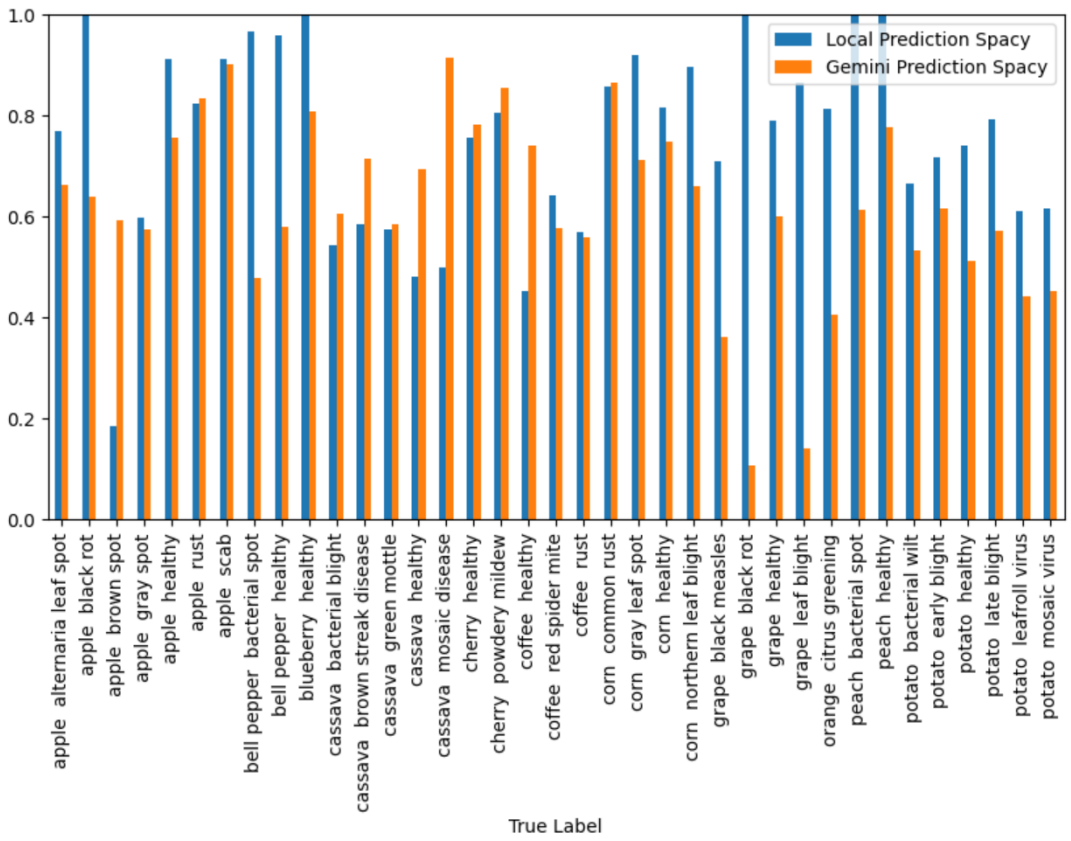

Figure 8: Comparative Accuracy of Local Prediction and Gemini Prediction Across Different Plant Diseases with spaCy, grouped-by plants with labels.

BERT analysis

Figure 9 demonstrates the fine-tuned local model displays a better performance acroess a variety of plant disease classification in categories such as blueberry, healthy, grape black rot, peach bacterial spot, peach healthy. In labels such as cassava mosaic disease and cherry powederry mildew, the diagnosis is less precise for the local model.

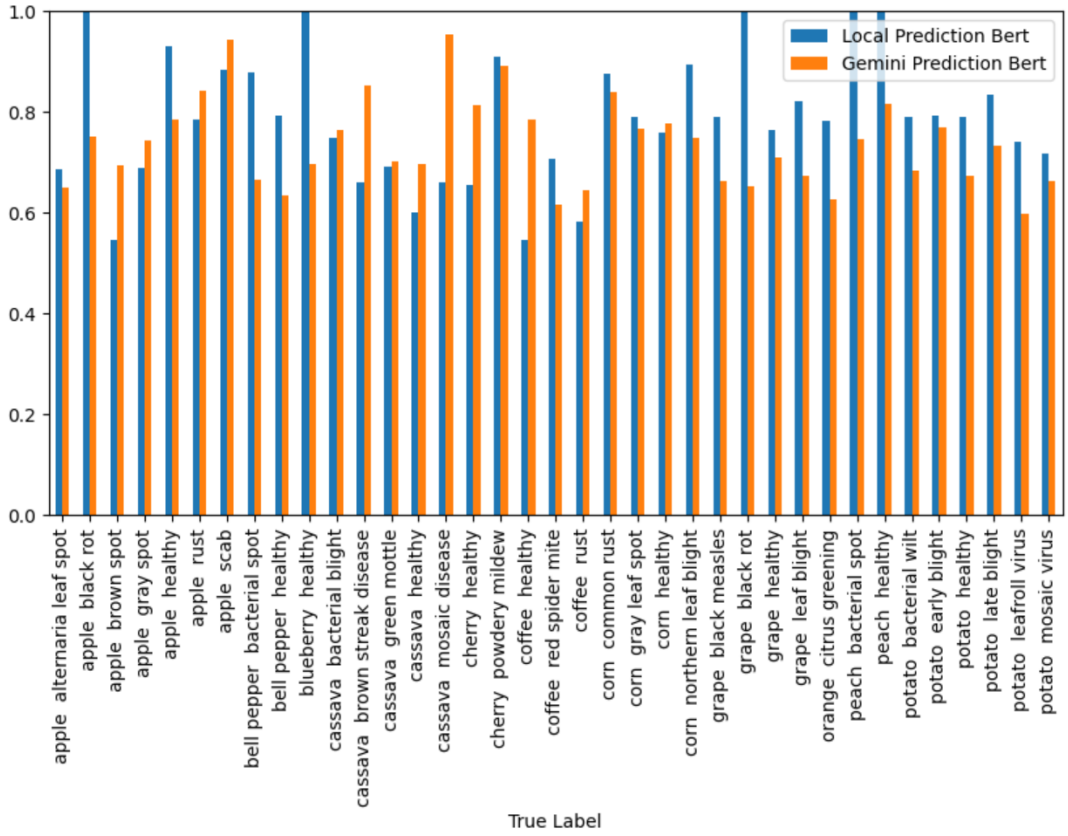

Figure 9: Comparative Accuracy of Local Prediction and Gemini Prediction Across Different Plant Diseases with BERT, grouped-by plants with labels.

3 Discussion

Based on the analysis of section 2.3 and section **??**, the Large Language Model (LLM) underperforms in most categories, in comparison to the fined-tuned local model, specifically in labels such as grape, black rot. It is possible due to the general purpose of the Large Language Model (LLM) that it isn't fine-tuned for plant disease classification in contrast with the plant disease dector local model.

4 Future Work

4.1 Image Classification

While the results favor the local model, the Large Language Model that is used for this study was Gemini 2.0 flash free tier which pose certain limitations in terms of accuracy and efficiency. The paid version of Gemini (2.5 Pro) or alternatives such as GPT-4.5, GPT-4o-latest may yield a different result.

The local model used in this study is only the fine-tuned version of a transformer-vison model, ViT-Base model with only 86M parameters, if the base model had more parameters such as the ViT-Huge with 632M parameters, and fine-tuned with a much more diverse dataset, the results would have been more interesting.

For future studies, consideration for the usage of more recent, advanced convolutional neural networks (CNNs) vision models such as EffectiveNet fine-tuned.

4.2 Evaluation

In this study, the current use of BERT-base and spaCy was used to measure the semantics similarity between the predicted and the true label. While these tools' efficacy is in captuing general linguistic similarity semantically, they rely on static embedding (spaCy), contextual embedding (BERT). Consequently, they may fail to capture the deeper relationship of domain-specific relationsip between the true label and predicted label. Future work may consider fine-tune BERT or derive a better model for evalution.

References

[1] H. Pallathadka, M. Mustafa, D. T. Sanchez, G. Sekhar Sajja, S. Gour, and M. Naved, "Impact of machine learning on management, healthcare and agriculture," *Materials today : proceedings*, vol. 80, pp. 2803–2806, 2023.

[2] Abhiram4, "PlantDiseaseDetectorVit2." `https://huggingface.co/Abhiram4/PlantDiseaseDetectorVit2`, 2023.

[3] Google, "vit-base-patch16-224-in21k." `https://huggingface.co/google/vit-base-patch16-224-in21k`, 2024.

[4] Google, "bert-base-uncased." `https://huggingface.co/google-bert/bert-base-uncased`, 2024.