Comparative Analysis of SIFT-KNN, Custom CNN, and ResNet-18 for Plant Leaf Disease Detection

Exploring traditional and deep learning models to advance precision in plant disease detection

Swaraj Bhanja
Department of Data Science and AI
Asian Institute of Technology
Pathumthani, Thailand
st125052@ait.asia

ABSTRACT

This study compares the performance of traditional machine learning methods against modern deep learning methods for plant leaf disease detection. A SIFT-KNN model is set as a baseline, while a custom CNN and a transfer-learning based ResNet-18 are attempted to be tuned against that benchmark. The aim is to compare each model in terms of visual disease pattern extraction. With the assumption that deep learning models will outperform traditional machine learning methods, model accuracy, generalizability and computational requirements are analyzed. The results offer ingenious insights into the strengths and weaknesses of each model in the domain of Computer Vision applied in agriculture.

1. INTRODUCTION

With a growing global population and demand for food, agriculture has become increasingly subject to loss of production due to plant diseases, threatening food security. Plant disease detection solutions tend to address food security and crop yield challenges by providing valuable insights and suggestions. However, traditional methods to identify and diagnose plant diseases require subject matter expertise which is labor intensive and maybe deficient in the developing parts of the world. Recent trends in Computer Vision and Machine Learning present solutions that can assist in early disease detection, which can be leveraged by farmers and argriculturists.

Computer Vision techniques enables models to understand visual data, allowing systems to identify patterns in plant leaves. Traditional approaches rely on manually selected features such as the Scale-Invariant Feature Transform (SIFT) which require less computational resources but suffer in terms of prediction accuracy due to their limitations in the ability to capture nuanced image features. On the other hand, deep learning models such as Convolutional Neural Networks (CNN) automatically learn intricate features from raw pixel data, allowing them to capture disease patterns that traditional models may have missed out on. Also, in the recent years, transfer learning methods with pre-trained models like ResNet, AlexNet, etc have grown in reputation offering a path to leverage knowledge gained from large-scale datasets like ImageNet, improving prediction accuracy significantly compared to the traditional methods like SIFT.

While deep learning models like Convolutional Neural Networks (CNN) have shown promising results in terms of high accuracy in the image classification domain, their computational architecture often hinders usage in low-resource environments. The plant disease detection domain tends to encapsulate high intra-class variability and intra-class similarity, suggesting that disease symptoms can differ significantly within a single disease type and maybe similar across different diseases. Thus, accurate prediction of disease type becomes challenging. Sophisticated CNN models require arrays of Graphics Processing Unit (GPU) in order to understand image properties for accurate predictions. This study aims to evaluate whether deep learning justifies its computational costs by comparing it to traditional feature-based machine learning models. By understanding the tradeoff between different approaches, the study aims to tap into a balance between accuracy and computational efficiency.

For this purpose of understanding the different Computer Vision techniques in detecting plant diseases, three research questions are presented:

RQ1: How does the accuracy of a traditional SIFT-KNN model compare to the custom CNN and ResNet-18 model for plant disease detection?
RQ2: Can a custom CNN perform comparable to Resnet-18 by using residual connections and batch normalization?
RQ3: What are the tradeoffs between accuracy and computational cost across models?

In an attempt to answer these questions, three initial hypotheses must be defined:

H1: ResNet-18 will achieve the highest accuracy due to its deep-seated architecture and pre-trained weights on ImageNet, surpassing both SIFT-KNN and the custom CNN.
H2: The custom CNN with residual connections and batch normalization will achieve accuracy comparable to ResNet-18.
H3: SIFT-KNN will demonstrate faster inference times and lower memory usage than deep learning models, making it suitable for low-resource regions.

This study attempts to contribute to the field of computer vision in agriculture by evaluating three distinct approaches to plant disease detection:
A. **SIFT-KNN** - A feature-based model that serves as a baseline and boasts efficient computation in resource-deficient settings.
B. **Custom CNN** - A deep learning model that leverages residual connections and batch normalization to achieve a balance between prediction accuracy and computational resource usage efficiency.
C. **ResNet-18** - A deep CNN pre-trained on large-scale data, expected to achieve remarkable accuracy due to advanced image processing capabilities but requiring more computational resources.

By comparing these approaches, this study aims to offer insights into the limitations of the three Computer Vision methods in the agriculture domain. This will enable informed decision-making regarding model selection to achieve the desired accuracy. This study may also serve as a foundation for deeper and more granular research in this domain, considering the possible solutions that may be developed in the long run.

2. RELATED WORK

Automated plant disease detection using Computer Vision has evolved from conventional methods like SIFT to sophisticated Deep Learning Techniques like CNN. Handcrafted techniques like SIFT were widely used in early image classification tasks. Phadikar et al. (2008) applied SVM with color and texture features for rice disease identification, while Arivazhagan et al. (2013) used KNN with color and texture-based descriptors for fruit disease detection, finding moderate success in limited setups **[1][2].**

With the development of CNN, Deep Learning revolutionized Computer Vision. Mohanty et al. (2016) demonstrated CNNs' capacity to extract complex disease patterns across crop types, achieving high classification accuracy on the PlantVillage dataset **[3]**. To enhance performance, Ferentinos (2018) extended this work by employing deep CNNs with pre-trained models, including AlexNet and VGG16 **[4]**. Residual Networks (ResNets), introduced by He et al. (2016), addressed the vanishing gradient problem, enabling the use of very deep networks and significantly advancing image classification accuracy **[5]**. ResNet models have found application in agriculture, as seen in Sun et al. (2020) for apple disease detection using ResNet-50, demonstrating high robustness and convergence rates **[6]**.

Despite the incredible accuracy offered by Deep Learning models in Computer Vision, researchers like Zhang et al. (2021) explored lightweight models for edge devices, using MobileNetV2 for real-time disease classification with fewer computational resources **[7]**. This study adds to this literature by comparing traditional feature-based SIFT-KNN with a custom CNN and ResNet-18, attempting to strike a balance between prediction accuracy and computational resource usage efficiency.

3. METHODOLOGY

3.1 Dataset Source

The **PlantVillage** dataset, containing over 20000 labeled images of healthy and diseased plant leaves, was used for the study. The dataset contains multiple disease types across different plant species, allowing for a powerful premise for robust model training and efficient predictions.

3.2 Data Preprocessing and Augmentation

The PlantVillage dataset was first split into the train, validation, and test folders in the 8:1:1 ratio for modeling, validation, and prediction purposes. Each image was then resized to 224x224. The extension of each image was converted from .JPG to .jpg to ensure ResNet and CNN models did not face any issues during image processing. For the CNN and ResNet-18 models, data augmentation techniques (rotation, flipping, and brightness adjustments) were applied to improve the generalization accuracy.

SIFT-KNN, which uses fixed descriptors, did not use data augmentation to preserve feature integrity.

3.3 Baseline Model: SIFT-KNN

An initial baseline model based on SIFT descriptors was used to extract features from grayscale images to create a baseline SIFT model. Each image was standardized to a fixed-length vector by truncating or padding the descriptors to a maximum of 500 and then flattened into 1D vectors. The K-Nearest Neighbors (KNN) classifier algorithm with **n_neighbors = 5** was chosen for classification. This approach resulted in an accuracy score of 17%.

```
predicted_labels = knn.predict(test_descriptors)
accuracy = accuracy_score(test_labels_encoded, predicted_labels)
print(f"SIFT-KNN Test Accuracy: {accuracy:.2f}")
print(classification_report(test_labels_encoded, predicted_labels, target_names=label_encoder.classes_))
```

Python

```
SIFT-KNN Test Accuracy: 0.17
                                            precision    recall  f1-score   support

                Pepper__bell___Bacterial_spot       0.25      0.02      0.04       100
                      Pepper__bell___healthy       0.12      0.93      0.22       148
                       Potato___Early_blight       0.33      0.09      0.14       100
                        Potato___Late_blight       0.13      0.13      0.13       100
                          Potato___healthy       0.08      0.62      0.14        16
                     Tomato_Bacterial_spot       0.41      0.15      0.21       213
                       Tomato_Early_blight       0.00      0.00      0.00       100
                        Tomato_Late_blight       0.57      0.02      0.04       191
                          Tomato_Leaf_Mold       0.00      0.00      0.00        96
                 Tomato_Septoria_leaf_spot       0.25      0.01      0.02       178
Tomato_Spider_mites_Two_spotted_spider_mite       0.25      0.09      0.13       168
                       Tomato__Target_Spot       0.12      0.02      0.04       141
     Tomato__Tomato_YellowLeaf__Curl_Virus       0.75      0.02      0.04       321
               Tomato__Tomato_mosaic_virus       0.67      0.05      0.10        38
                           Tomato_healthy        0.23      0.70      0.34       160

                                 accuracy                           0.17      2070
                                macro avg       0.28      0.19      0.11      2070
                             weighted avg       0.33      0.17      0.11      2070
```

To explore the scope of improvement, SIFT descriptors were replaced with raw pixel values of the grayscale images, flattened into 1D vectors, and used directly as input features for the K-Nearest Neighbors (KNN) classifier. This approach achieved 37% accuracy, highlighting the limitations of traditional handcrafted features in capturing complex disease patterns.

```
print("KNN Model with Best Parameters Accuracy:", accuracy_score(test_labels_encoded, predicted_labels))
print(classification_report(test_labels_encoded, predicted_labels))
```

Python

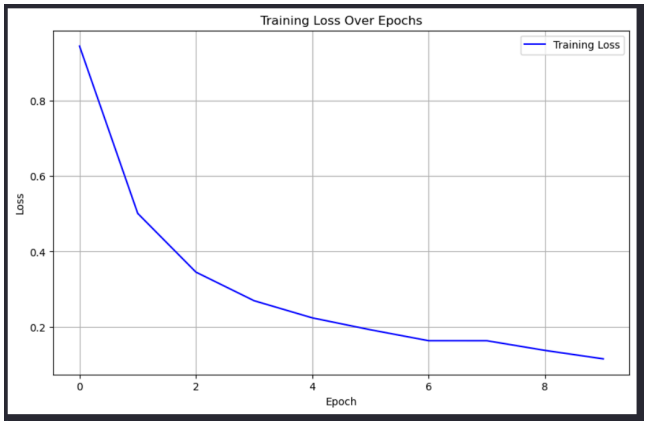
```
KNN Model with Best Parameters Accuracy: 0.3681159420289855
              precision    recall  f1-score   support

           0       0.69      0.09      0.16       100
           1       0.55      0.61      0.58       148
           2       0.80      0.16      0.27       100
           3       0.24      0.50      0.32       100
           4       0.60      0.12      0.19        16
           5       0.30      0.76      0.43       213
           6       0.33      0.01      0.02       100
           7       0.35      0.47      0.40       191
           8       0.38      0.47      0.42        96
           9       0.24      0.34      0.29       178
          10       0.61      0.30      0.41       168
          11       0.15      0.03      0.05       141
          12       0.52      0.34      0.41       321
          13       1.00      0.05      0.10        38
          14       0.42      0.43      0.42       160

    accuracy                           0.37      2070
   macro avg       0.47      0.31      0.30      2070
weighted avg       0.43      0.37      0.34      2070
```

It also depicts that pixel-level information of images can also provide meaningful signals for primary classification when processed effectively. It also sets the ground for deep learning techniques that capture nuanced plant leaf disease patterns.

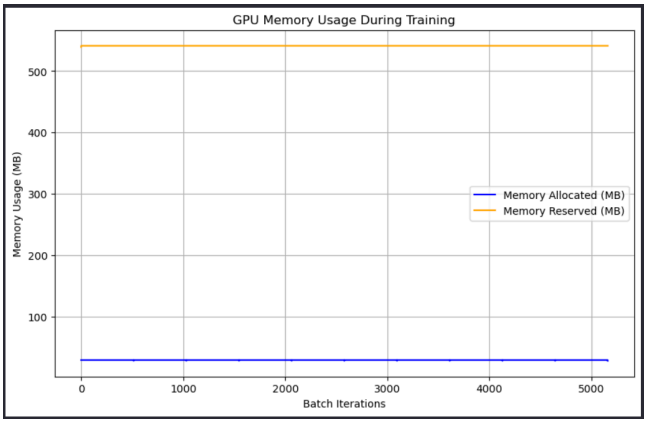
3.4 Custom CNN

The custom CNN was created by introducing residual connection and batch normalization. The model contains four convolutional blocks, each representing two layers followed by batch normalization, ReLU activation function, and max-pooling. Skip connections were added to each block to improve the gradient flow and tackle the vanishing gradient problem. Fully connected layers were replaced by Global Average Pooling to reduce model parameters and enhance generalization. To reduce computational overhead, the number of filters was scaled down, starting with 32 filters in the first block and progressively increasing to 256 filters in the final block. Additionally, 1x1 pointwise convolutions replaced one of the 3 x 3 convolutions in each block, significantly reducing the number of parameters while retaining feature extraction capacity. The model was trained with mixed precision using PyTorch AMP, leveraging lower precision for faster computations and reduced memory usage. To further optimize training time, images were resized to a lower resolution of 112 x 112 during initial training, with the option to fine-tune at higher resolutions. The model was trained over 10 epochs with Adam Optimizer with a learning rate 0.001. With an accuracy of 95% appreciably comparable to that of ResNet-18, a resource usage of around 40 MB VRAM, and a training time of 14.28 minutes, the custom CNN attempts to provide an alternative to pre-trained models like ResNet-18, aiming to strike a balance between prediction accuracy and computational resource usage efficiency.

The following shows the training loss of the custom CNN:



The following shows the usage of VRAM during the training process for the custom CNN:



The following shows the time elapsed for training and the test accuracy of the custom CNN:

```
end = torch.cuda.Event(enable_timing=True)
end.record()

torch.cuda.synchronize()
print(f"Total Time (in mins) elapsed: {(start.elapsed_time(end) / 60000):.2f}")
```

```
Total Time (in mins) elapsed: 14.28
```

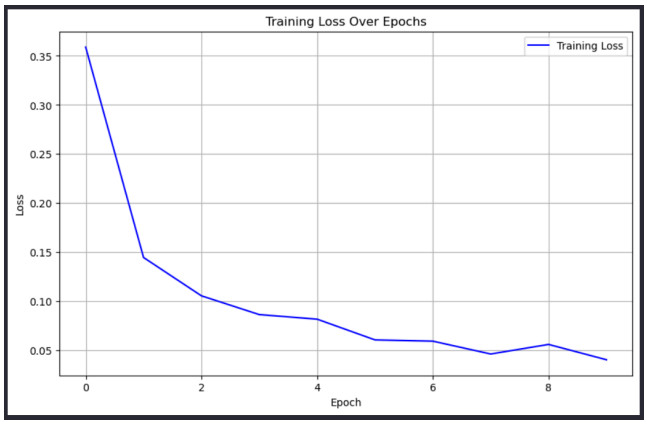
```
model.eval()
all_labels = []
all_preds = []
with torch.no_grad():
    for images, labels in test_loader:
        images, labels = images.to(device), labels.to(device)
        with torch.amp.autocast('cuda'):
            outputs = model(images)
        _, predicted = torch.max(outputs, 1)
        all_labels.extend(labels.cpu().numpy())
        all_preds.extend(predicted.cpu().numpy())

accuracy = accuracy_score(all_labels, all_preds)
print(f"Custom CNN Test Accuracy: {accuracy:.2f}")
print(classification_report(all_labels, all_preds, target_names=train_dataset.classes))
```

```
Custom CNN Test Accuracy: 0.95
```

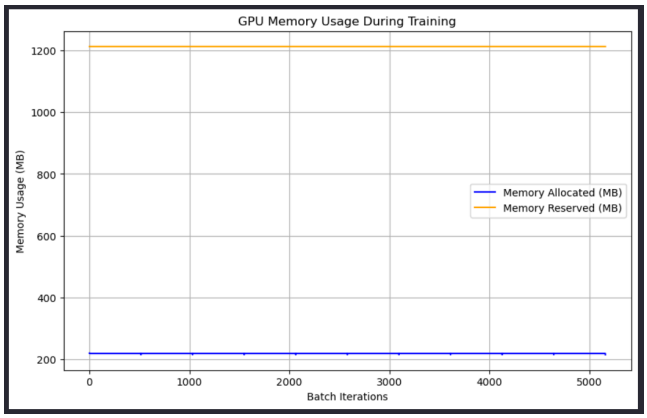
3.5 ResNet-18 with Transfer Learning

ResNet-18 pre-trained on ImageNet was implemented with weights as IMAGENET1K_V1. In this model, the final fully connected layer was replaced to match the number of classes (12) in the PlantVillage dataset. The pre-trained weights were used to tap into ResNet-18's robust and powerful feature extraction capabilities. Images were resized to 224x224, which is a requirement of ResNet-18. They were also converted to tensors, with data normalization being applied. ResNet-18 is expected to achieve the highest prediction accuracy in recognizing diverse disease patterns. ResNet-18 achieved an accuracy of 96.33% a resource usage of around 220 MB VRAM, and a training time of 27.54 minutes. This establishes the ground that the custom CNN, which strikes a balance between accuracy and efficiency in terms of training and computational resource usage.

The following shows the training loss of the ResNet-18 model:



The following shows the usage of VRAM during the training process for ResNet-18:



The following shows the time elapsed for training and the test accuracy of the ResNet-18 model:

```
end = torch.cuda.Event(enable_timing=True)
end.record()

torch.cuda.synchronize()
print(f"Total Time (in mins) elapsed: {(start.elapsed_time(end) / 60000):.2f}")
```

```
Total Time (in mins) elapsed: 27.54
```

```
resnet18.eval()
correct = 0
total = 0
with torch.no_grad():
    for images, labels in test_loader:
        images, labels = images.to(device), labels.to(device)
        outputs = resnet18(images)
        _, predicted = torch.max(outputs.data, 1)
        total += labels.size(0)
        correct += (predicted == labels).sum().item()

accuracy = 100 * correct / total
print(f'ResNet Model Test Accuracy: {accuracy:.2f}%')
```

```
ResNet Model Test Accuracy: 96.33%
```

4. RESULTS AND EVALUATION

Accuracy, denoting the number of test instances labeled correctly by the models, was used to evaluate prediction quality. Inference Time and Memory Usage were also used to compute computational resource usage efficiency. **SIFT-KNN:** Achieved a satisfactory accuracy of 37%, with

relatively low memory usage and high inference speed, supporting Hypothesis 3.

Custom CNN: Achieved an accuracy appreciably comparable to that of ResNet-18, supporting Hypothesis 2. Skip connections and batch normalizations were helpful and effective in capturing complex image features.

ResNet-18: Achieved the highest accuracy of 96.33%, confirming Hypothesis 1, but was computationally the most expensive among all the three models.

The following image depicts a summary of the performance of the models:

Model	Accuracy	Inference Time	Memory Usage (MB)	Remarks
SIFT-KNN	37%	Fast	Minimal	Efficient in resource use but limited in capturing complex patterns.
Custom CNN	95%	Faster than ResNet	40	Achieved competitive accuracy with significantly lower resource demands.
ResNet-18	97%	Slower than Custom	220	Superior accuracy but computationally expensive in time and memory.

5. DISCUSSION AND ANALYSIS

5.1 Analysis

The custom CNN seems to provide a middle-ground solution between SIFT-KNN's high computation resource usage efficiency and ResNet-18's high prediction accuracy. The custom CNN's residual connections and batch normalization significantly improved accuracy, suggesting that network depth with careful regularization can yield near state-of-the-art performance without needing a full ResNet.

5.2 Trade-Offs in Model Selection

For resource-deficient settings, SIFT-KNN is practical due to its low memory usage and inference speed. However, Resnet-18 or the custom CNN should be the way forward when the stakes are high on accurate predictions. The custom CNN captures the essence of bridging the gap between traditional handcrafted Feature Selection and Deep Learning methods by incorporating modern techniques without compromising computation resource usage efficiency.

5.3 Implications for Real-World Applications

The custom CNN and ResNet-18 models have the potential to be used in agricultural settings where computational resources are abundant, offering high accuracy in plant disease detection. SIFT-KNN could be applied in constrained environments, providing actionable insights in low-resource regions.

6. CONCLUSION

This study contributes to computer vision in agriculture by comparing traditional feature-based and deep-learning methods for plant disease detection. These insights can guide the development of scalable and accessible disease detection systems across diverse agricultural contexts.

ACKNOWLEDGMENTS

The author thanks Dr. Cherdsak Kingkan and Mr. Biplav Regmi for providing insights and checking the work throughout the study.

REFERENCES

[1] Phadikar, S., Sil, J., & Das, A. K. (2008). Rice disease identification using pattern recognition techniques. In Proceedings of the IEEE 11th International Conference on Computer and Information Technology (ICCIT), pp. 420-423. https://doi.org/10.1109/ICCIT.2008.236

[2] Arivazhagan, S., Shebiah, R. N., Ananthi, S., & Varthini, S. V. (2013). Detection of unhealthy region of plant leaves and classification of plant leaf diseases using texture features. *Agricultural Engineering International: CIGR Journal*, 15(1), 211-217.

[3] Mohanty, S. P., Hughes, D. P., & Salathé, M. (2016). Using deep learning for image-based plant disease detection. *Frontiers in Plant Science*, 7, 1419. https://doi.org/10.3389/fpls.2016.01419

[4] Ferentinos, K. P. (2018). Deep learning models for plant disease detection and diagnosis. *Computers and Electronics in Agriculture*, 145, 311-318. https://doi.org/10.1016/j.compag.2018.01.009

[5] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770-778. https://doi.org/10.1109/CVPR.2016.90

[6] Sun, H., Fu, Y., Liu, H., & He, Y. (2020). Apple lesion detection in orchards based on deep learning methods of cyclical CNN and Faster R-CNN. *IEEE Access*, 8, 3304-3312. https://doi.org/10.1109/ACCESS.2019.2962457

[7] Zhang, Q., Xu, X., & Zheng, G. (2021). Lightweight deep learning model for plant disease detection on edge devices. *Agricultural Engineering International: CIGR Journal*, 23(1), 101-109.