Enhancing Potato Crop Health: CNN and Deep Learning Approaches for Early Blight and Late Blight Detection: CS 7643

Rupeshsaran Govindan Balamurugan
Georgia Institute Of Technology
Atlanta, GA 30332
rbalamurugan6@gatech.edu

Abstract

Advancements in farming using deep learning are particularly valuable because visual assessments are commonly used to distinguish healthy crops from those affected by disease. In this report, I explore the application of deep learning techniques for detecting and classifying potato leaf diseases, with a focus on Early Blight and Late Blight. We will explore this problem with a multi layered CNN image recognition model,to tackle the challenges of accurate long-term disease prediction. Through a detailed experimental setup, I document the implementation challenges, describe the methodology, architecture, layers and present accurate results and plot of both accuracy and loss of the final model. My findings reveal the superior performance of my approach in predicting disease progression over extended periods as compared to the traditional methods used today, providing valuable insights into potential applications in agricultural technology and crop management strategies. This study not only enhances our understanding of plant disease dynamics but also establishes a new benchmark in the precision farming technologies sector.

1. Introduction/Background/Motivation

Potatoes are a crucial agricultural product worldwide and represent a significant income source for farmers. Unfortunately, they are prone to various diseases that can drastically affect both yield and quality, with leaf diseases being particularly damaging and the most deadly and common disease [9]. Early and accurate detection of these diseases is critical for effective management and control. Traditionally, disease diagnosis has relied on visual inspections, which are subjective and labor-intensive. Recently, there's been increasing interest in applying deep learning techniques to automate the detection and classification of these diseases, promising more efficient and accurate diagnostics.

The potato, Solatium tuberosum is the fourth-ranked food crop used to feed an increasing global population because of its adaptability in cultivars and high complex carbohydrate content[2]. Currently, the detection and diagnosis of potato leaf diseases primarily rely on manual visual inspections by farmers or agricultural experts. This traditional method is subjective and varies greatly in accuracy depending on the observer's experience. It can also be time-consuming and inefficient, especially for large-scale operations[5]. Additionally, these manual inspections are not scalable and can lead to delayed responses to disease outbreaks, potentially resulting in significant crop damage and yield loss. The need for timely and accurate disease identification highlights the limitations of current practices in handling plant health effectively.

Successful detection and management of potato diseases, such as Early Blight and Late Blight, are crucial not only for the agricultural sector but also for global food security. If the development and application of Convolutional Neural Network (CNN) models for detecting these diseases are successful [14], Early and accurate detection of potato diseases allows for timely intervention, preventing the spread and minimizing damage. This can significantly increase crop yields and quality, directly benefiting farmers economically and contributing to a more stable food supply.[6]. Diseases like Early Blight and Late Blight can cause severe economic losses. By reducing crop losses, the technology not only saves money but also boosts profitability for farmers , contributing to the economic stability of the communities that depend on agriculture as their primary source of income [3]. The ability of CNNs to "reduce the subjectivity that comes with naked-eye observation" as mentioned by Singh et al. (2015), allows for more precise and less labor-intensive monitoring of crop health, which can lead to substantial cost savings.[6].

With the ability to accurately identify disease symptoms early, farmers can implement targeted interventions sooner, potentially saving large portions of their crops from damage. This proactive approach to disease management not only helps secure the livelihood of farmers but also con-

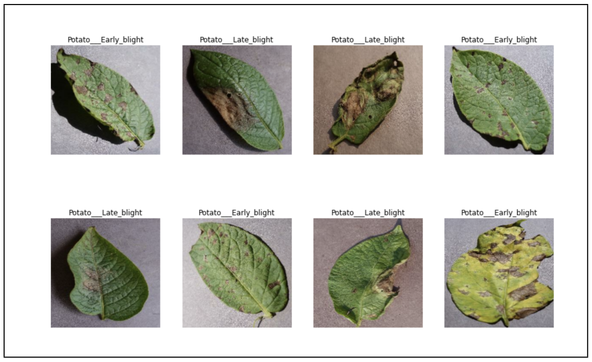

Figure 1. Different images of potato leaves in different stages of blight

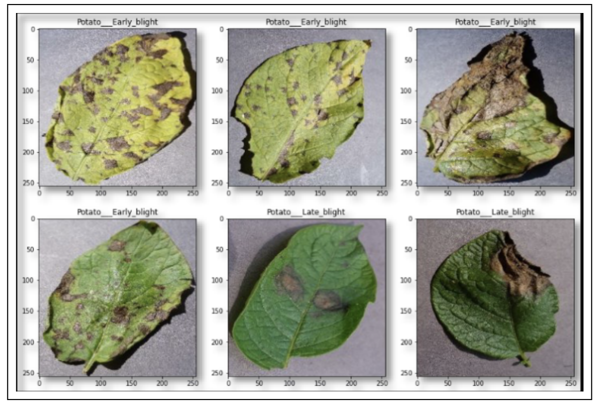

Figure 2. Another set of potato leaves in different stages of blight

tributes to food stability in regions heavily dependent on agriculture.

Furthermore, the adoption of deep learning techniques such as CNNs for disease detection promotes the integration of more scientific approaches in agriculture, fostering innovation and technological advancement. This shift towards high-tech agriculture could lead to broader changes in farming practices, encouraging more farmers to embrace data-driven decision-making [1]. By reducing reliance on traditional, often inefficient methods of disease detection, such as manual inspection, farmers can optimize their use of resources, including labor and agricultural inputs, leading to more sustainable farming practices and better environmental stewardship [12].

For my dataset I utilized the Plant Village dataset, which is publicly available on Kaggle and curated by Priyak. This dataset is comprehensive, containing approximately 20,600 images across various plant types, including 2,152 images specifically of potato leaves. These potato leaf images are further classified into three categories representing different states of health: Healthy, Early Blight, and Late Blight. The dataset is organized in JPEG format, which is standard for image processing tasks.

The dataset's distribution includes 1,000 images each for Early Blight and Late Blight, while the Healthy category contains 152 images. This distribution allows for a balanced training approach for the disease categories but presents a challenge in the under-representation of healthy leaf images[16]. To address potential data imbalance and to enhance the robustness of the model, I employed data augmentation techniques. These techniques included basic geometric transformations such as rescaling, horizontal and vertical flipping, and random rotation of 0.2 degrees. This approach helps in artificially expanding the dataset, particularly beneficial for the underrepresented class, ensuring that the deep learning model learns to generalize well across varied data inputs.

2. Approach

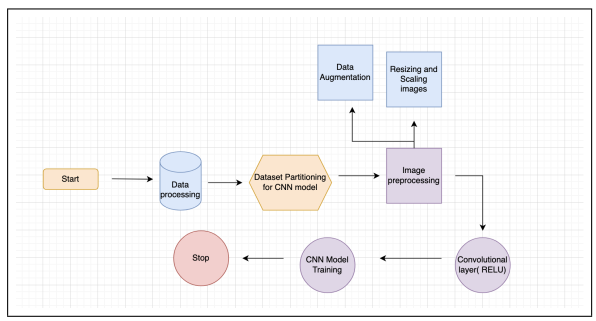

Figure 3. Basic Architecture

The idea is to use a CNN model that processes images of potato leaves to detect signs of diseases such as Early Blight and Late Blight. The model architecture includes multiple convolutional layers, each followed by subsampling layers, which help in extracting and reducing the feature dimensionality while retaining the important information necessary for classification.Convolutional Neural Networks (CNNs) are a widely favored choice in deep learning. Several research studies have utilized CNNs to identify plant diseases by analyzing the health of the leaves[5]. Convolutional Neural Networks (CNNs) typically consist of several multi-layered convolutional layers organized by their functions.

Following these, one or more fully connected layers usually succeed a subsampling layer within the network structure. The input to each subsequent feature layer comprises a compact area of features derived from the previous layer [15]. CNNs are highly proficient in handling computer vision tasks. Unlike Artificial Neural Networks (ANNs), CNNs have the advantage of parameter sharing, which effectively reduces the total number of parameters required for the model [7].

After the image processing was carried out the images were then fed into the CNN model in order to carry out the training process, The initial results were then used to further enhance the hyper parameters and make sure they are present within appropriate range for producing the final results. The final batch size was 32 and the images were all of 256 x 256 size.

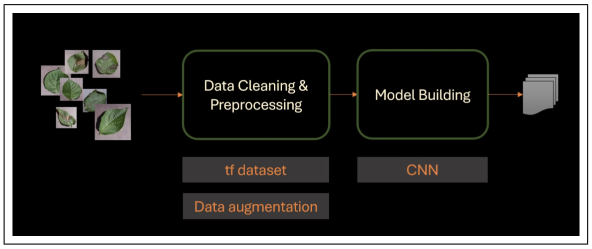

Figure 4. Basic Idea of how the model works

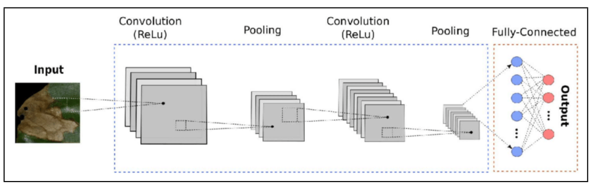

Figure 5. Depiction of the CNN neural network model used to identify leaf blight

2.1. Data Preprocessing

The project was initiated by by assembling a dataset consisting of 2,152 images of potato leaves as mentioned in figure 1, which were meticulously categorized into three distinct classes: Healthy, Early Blight, and Late Blight. These images were sourced from the comprehensive Plant Village dataset, a well-known repository used extensively for agricultural research. To ensure that the dataset was suitable for effective processing by our Convolutional Neural Network (CNN), a detailed preprocessing stage was undertaken[7].

During preprocessing, each image was standardized to ensure uniformity in size and format across the dataset. This standardization process involved resizing all images to a consistent resolution of 256x256 pixels, a common practice that helps optimize computational efficiency and model performance. Additionally, the image files were converted into a uniform JPEG format, streamlining the input process for the CNN.

The dataset is then split into training, validation and testing sets.Typically, around 70-80 percent of the data is used for training, with 10-15 percent is reserved for validation to tune hyperparameters and mitigate overfitting. After the image processing was carried out the images were then fed into the CNN model in order to carry out the training process, The initial results were then used to further enhance the hyper parameters and make sure they are present within appropriate range for producing the final results.

2.2. CNN Architecture Layers

2.2.1 Convolutional layers

Convolutional layers are the core building blocks of Convolutional Neural Networks (CNNs). These layers use filters, also known as kernels, to extract features from the input images through a process called convolution. The operation can be mathematically represented as follows:

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n) \cdot K(i-m, j-n)$$

where I is the input image, K is the kernel, and S is the feature map produced by the convolution. In this formula:

- $I(m,n)$ represents the pixel values of the input image.
- $K(i-m, j-n)$ represents the kernel that slides over the image.
- (i,j) are the coordinates of the output feature map.

Each convolutional layer applies multiple filters to the input image to produce a set of feature maps. These maps highlight different aspects of the image, essential for understanding complex features like edges, corners, and textures[4]. The depth of the feature map is equal to the number of filters used.

2.2.2 Sub-Pooling layers

Pooling layers are integral components of Convolutional Neural Networks (CNNs) that follow the convolutional layers. Their primary function is to reduce the spatial dimensions (i.e., width and height) of the incoming feature maps[11]. This reduction is crucial as it decreases the computational load for subsequent layers and helps in mitigating the risk of overfitting by providing an abstracted form of the features.

The operation performed by a pooling layer can be expressed in general terms as follows:

$$P(i,j) = \max_{m,n \in \mathcal{W}} F(m,n)$$

where:

- $P(i,j)$ is the output of the pooling operation at position (i,j).
- $F(m,n)$ represents the elements of the feature map within the window $\mathcal{W}$, which is typically a small region like 2×2 or 3×3.

- The max function is used for Max Pooling, which takes the maximum value within the window as the output. Alternatively, Average Pooling can be used, which computes the average of the values in $\mathcal{W}$.

Two primary types of pooling are commonly employed:

- **Max Pooling**: Selects the maximum element from the region of the feature map covered by the filter. This is the most commonly used pooling method in CNNs as it captures the most prominent features, contributing significantly to the model's robustness against variations in the input data [11].
- **Average Pooling**: Calculates the average of all elements in the filter region, providing a smoothed feature map output.

2.2.3 Fully-Connected Layers

These layers connect the output of the pooling layers to a final classification layer. They synthesize the features extracted by previous layers to form high-level representations of the input images.

2.2.4 Final Softmax Output layer

A softmax activation function is used in the final layer to provide a probability distribution over the three classes, indicating the likelihood of each disease state.

2.3. CNN Architecture Layers

2.4. Problems Encountered

Over fitting: The model was initially over fitting which is not that rare for CNN models. The solution was in this case to use batch normalization that allowed each layer to learn independently, reducing overfitting.

High Variance in Initial Tests: The initial model was over fitting, as indicated by high accuracy on training data but poor performance on validation data. This was partly due to the complexity of the model and insufficient regularization.

Initial Model Configuration: The very first configuration of the CNN did not perform as expected. While it was capable of identifying features within the training set, it failed to generalize these findings to the validation set, which led to disappointing initial results.

As mentioned above one of the very first problems that I anticipated was the initial code repository model not working properly due to the amount of layering present in the model, however I was not able to simply overcome this by adjusting the layers Hence the very first thing that I tried did not work at all, which was adjusting the layers ord adding more RELU layers. I then experimented with reducing the number of layers and altering the size of the convolution filters to find a balance that maintained learning capability while avoiding over fitting and this was the solution that worked.

3. Experiments and Results

The model is a Convolutional Neural Network (CNN) that processes images through multiple convolutional layers, each designed to detect distinct features in the input images. The model starts with the first convolutional layer (C1), where we apply 32 filters of size 3x3 to the input image, which has varying dimensions [10]. This layer uses valid padding, meaning it does not artificially increase the size of the image with padding, resulting in feature maps that are slightly smaller than the input. This layer outputs 32 feature maps of size 254x254, each representing different features detected in the image, such as edges and textures.

Following the first layer, the network includes a series of additional convolutional layers (C2 to C5), each paired with a subsampling layer that performs max-pooling with a 2x2 window. These layers progressively reduce the spatial dimensions of the feature maps while increasing their depth, enhancing the network's ability to capture complex patterns [13]. For instance, the second convolutional layer (C2) uses 64 filters to produce 64 feature maps of size 3x3, which are then pooled to 125x125 feature maps. This process of convolution followed by max-pooling continues through additional layers, each time abstracting the image features to a higher level and reducing their dimensionality to make the model computationally efficient and robust against overfitting.[8]

Figure 6. Training validation accuracy and data plots

In the plots shown above we can see the training of the validation accuracy and the validation data set. While training there was wide fluctuations when it came to changing epochs but the learning rate proved to be fine. After about 20 epochs with a batch size of 32. predicted the average accuracy of the validation data was found to be **95.37655** percent. In figure 7 we can see another validation accuracy and validation loss plot that was done with about 50 epochs but with the same batch size learning rate.

Table 1. Convolutional Layers in the CNN Model

Layer	Filter Size	Feature Map Size
C1	3x3	254x254
C2	3x3	125x125
C3	3x3	60x60
C4	3x3	14x14
C5	3x3	3x3

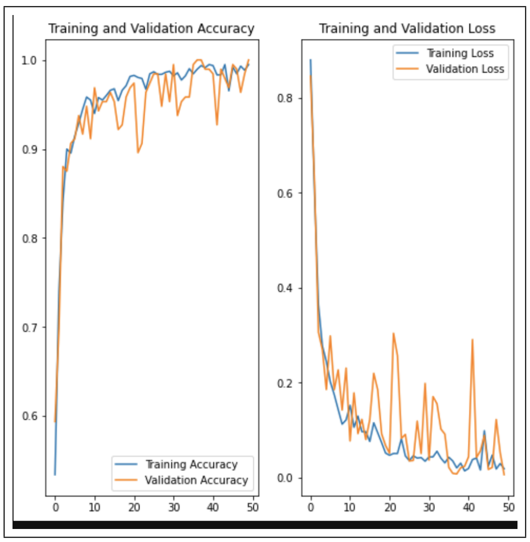

Figure 7. Training validation accuracy and data plots

After the modifications started producing good enough results the images were sent through the model in order to predict what stage of blight the leaf was going through. and the results are shown below in figure 8. The blight detection is then reflected by a confidence percentage on how sure the model is from its previous training on being correct when differentiating between early and late stages of blight.

4. Conclusion and Prospects for the future

In this project, the classification of potato diseases was tackled using a Convolutional Neural Network (CNN). The structure of the problem, involving the analysis of visual data (images of potato leaves) to identify the type and presence of disease, perfectly suited the capabilities of CNNs. The CNN architecture reflected the complexity of the task by incorporating multiple convolutional layers designed to

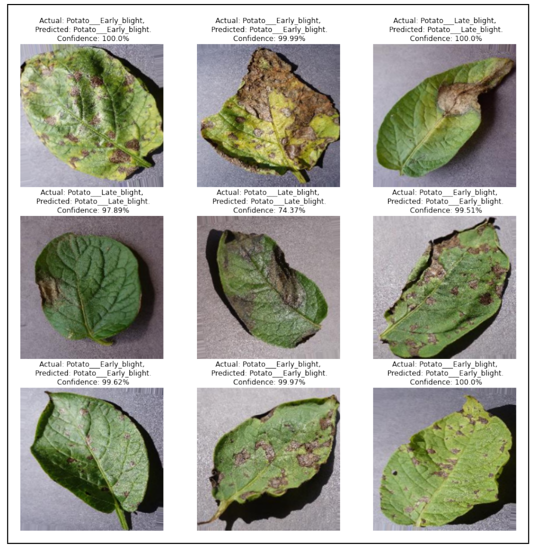

Figure 8. Final results

extract features from the images at various levels of abstraction. These were followed by pooling layers that reduced data dimensionality, enhancing computational efficiency and helping to mitigate the risk of overfitting.

The model included layers with learned parameters—specifically, the convolutional and the fully connected layers at the end of the network. These layers were essential for extracting and interpreting features. Conversely, the parts of the model without learned parameters included the pooling layers and the softmax activation function, which were used to classify the output into distinct categories based on the extracted features.

Data handling involved expecting images as inputs, which underwent pre-processing to normalize pixel values and resize them to uniform dimensions. Post-processing involved using the softmax function to convert logits from the last layer into probability scores, facilitating the interpretation of results. The categorical cross-entropy loss function was selected, ideal for multi-class classification tasks as it quantifies the disparity between predicted probabilities and the actual distribution.

The performance of the model was rigorously evaluated to check for overfitting. Initially, some evidence suggested that the model might be learning excessively from the training data. However, adjustments such as introducing dropout layers and enhancing data augmentation techniques helped mitigate this issue, leading to improved generalization on unseen data.

Hyperparameters like the number of layers, filter sizes, learning rate, and number of epochs were tuned based on experimental results, using the Adam optimizer known for its efficient handling of sparse gradients and adaptive learning rate capabilities. TensorFlow was chosen for developing

Student Name	Contributed Aspects	Details
Rupeshsaran Bala	Data processing and Implementation	Intital data processing in order to maintain uniformity among all the images.
Rupeshsaran Bala	Image pre-processing	Pre processed the images to fit the 32 batch size 256x256 size requirements..
Rupeshsaran Bala	CNN Model implementation	CNN model implementation from previous exisitng code in order to better fit for training purposes.
Rupeshsaran Bala	Initial Layer-Model configuration adjustments	Scraped the dataset for this project and trained the CNN of the encoder.
Rupeshsaran Bala	Report writing and editing	Wrote the entire report and edited it with accurate results from the models and code.
Rupeshsaran Bala	Accuracy and loss plot model training	Trained and validated the CNN model with accurate results.

Table 2. Contributions of team members.

the model due to its comprehensive library and supportive community for deep learning applications.

The foundation for the model was built upon existing TensorFlow tutorials and models that demonstrated similar tasks, which helped understand the necessary components and configurations for effectively dealing with image data.

Future work could expand on this foundation in several ways. Exploring more sophisticated CNN architectures such as ResNets or DenseNets could potentially capture deeper or more subtle features in the images. Unsupervised or semi-supervised learning methods could leverage unlabeled data, which is often plentiful in agricultural applications. Another promising direction involves deploying these models in real-world scenarios, integrating them into mobile applications for use directly in fields. This would provide real-time data and analysis, helping farmers manage crop health more effectively and promptly.

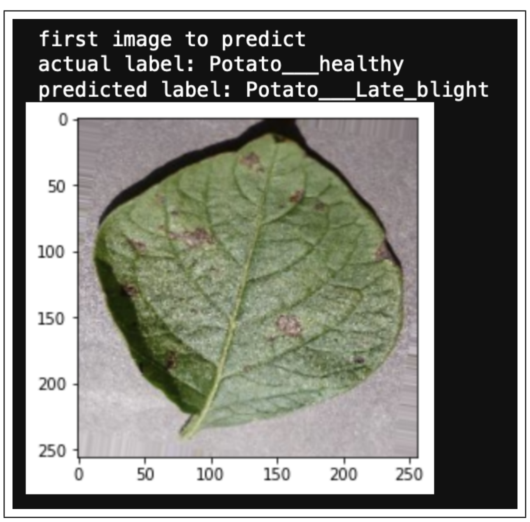

Figure 9. Image of model identifying stages of blight

5. Work Division

The final work division does not really matter in my case as its an individual project that did not include any other teamates other than myself in it. However I have added the different efforts that went into this paper and have added myself as the sole participant in the formatting and creation of this report and data.

I would also like to add that I formally requested and had prior permission before making this project individually by myself. I made a private piazza post and also talked to the professor in person about my situations and how I could not find teammates for this project and hence the reason why I am doing an individual project.

References

[1] A. Risnumawan A. Sholihati, I. A. Sulistijono and E. Kusumawati. Potato leaf disease classification using deep learning approach. *https://doi.org/10.1109/ies50839.2020.9231784*, 2020. 2

[2] Tijjani Ahmadu A. E. Abdullahi and K. Ahmad. The role of crop protection in sustainable potato (solanum tuberosum l.). *Production toAlleviate Global Starvation Problem: An Overview*, 15, 2021. 1

[3] C. Rodríguez C. Sullca, C. Molina and T. Fernández. Diseases detection in blueberry leaves using computer vision and machine learning techniques. *https://doi.org/10.18178/ijmlc.2019.9.5.854*, 2019. 1

[4] C. Rodríguez C. Sullca, C. Molina and T. Fernández. seases detection in blueberry leaves using computer vision and machine learning techniques. *https://doi.org/10.18178/ijmlc.2019.9.5.854*, 2019. 3

[5] H. Uga S. Kagiwada E. Fujita, Y. Kawasaki and H. Iyatom. Basic investigation on a robust and practical plant diagnostic system.

https://doi.org/10.1109/ICMLA.2016.0178, 2016. 1, 2

[6] Singh et al. (2015). discuss how automated disease detection reduces the subjectivity and laborintensive aspects of traditional methods, enhancing efficiency and accuracy). *https://ieeexplore.ieee.org/document/7164858*, 15, 2021. 1

[7] Md. A. Iqbal and K. H. Talukder. Detection of potato disease using image segmentation and machine learning. *https://doi.org/10.1109/wispnet48689.2020.9198563*, 2020. 2, 3

[8] J. Théau J. Boulent, S. Foucher and P.-L. St-Charles. Convolutional neural networks for the automatic identification of plant diseases. *https://doi.org/10.3389/fpls.2019.00941*, 2019. 4

[9] Kreuze J.F. Souza-Dias J.A.C. Jeevalatha A. Figueira A.R. Valkonen J.P.T. Viral diseases in potato, 2020. https://doi.org/10.1007/978-3-030-28683-5$_1$1. 1

[10] S. Ren K. He, X. Zhang and J. Sun. Keep residual learning for image recognition. *https://doi.org/10.1109/cvpr.2016.90*, 2016. 4

[11] K. Wahid M. Islam, Anh Dinh and P. Bhowmik. Detection of potato diseases using image segmentation and multiclass support vector machine. *https://doi.org/10.1109/CCECE.2017.7946594*, 2017. 3, 4

[12] D. P. Hughes S. P. Mohanty and M. Salathé. Using deep learning for image-based plant disease detection. *https://doi.org/10.3389/fpls.2016.01419*, 2016. 2

[13] Varsha V. Singh and A. K. Misra. Detection of unhealthy region of plant leaves using image processing and genetic algorithm. *https://doi.org/10.1109/icacea.2015.7164858*, 2015. 4

[14] V. Singh Varsha and A. K. Misra. Detection of unhealthy region of plant leaves using image processing and genetic algorithm. *https://ieeexplore.ieee.org/document/7164858*, 2015. 1

[15] Wikipedia. Training, validation, and test data sets," wikipedia, jun. 07, 2019. *https://en.wikipedia.org/wiki/Training,*$_validation,_and_test_data_sets$, 2019.2

[16] Y. Bengio Y. LeCun and G. Hinton. Deep learning," nature review articles, vol. 521, no. 7553, pp. 436–444, may 2015. *https://doi.org/10.1038/nature14539*, 2015. 2

5.1. Code repositories

The code repository I used is a previously implemented version of the leaf blight detection algorithm and deploys a simple CNN layer approach to this problem set. Some of the notable changes I made to the original code repository are:

- The pre-processing steps were enhanced to include more sophisticated techniques for image normalization and noise reduction.
- Significant effort was put into tuning the hyperparameters, including learning rate, batch size, and the number of epochs.
- To address the issue of data imbalance and to increase the robustness of the model against variations in image data, the data augmentation strategies were expanded. Techniques such as rotations, scaling, and color adjustments were introduced, enhancing the model's ability to generalize from the training data to real-world scenarios where lighting and background conditions may vary.
- Custom loss functions and new evaluation metrics were introduced to better reflect the practical goals of the project, such as maximizing the accuracy of detecting specific types of blight most detrimental to crop yield.
- The architecture of the CNN was refined to better capture the nuances of leaf blight symptoms. This involved adjusting the number and size of convolutional layers and filters to optimize the detection capabilities.

  Link to original code repository: https://github.com/codebasics/potato-disease-classification/blob/main/training/tf-lite-conversion-post-training.ipynb