HarvestNet - Vegetable and Fruit Recognition using CNN - Group I

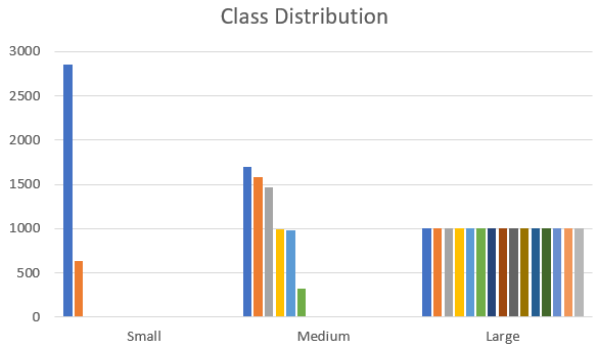
A. Abstract

The aim of the project is to provide a comprehensive analysis on vegetable and fruits classification across three datasets that vary by number of classes on three models - ResNet18, VGGNet16 and AlexNet. Data transformation and augmentation is performed to remove class imbalance and improve intra-class diversity. Models are compared based on precision, recall, F1-score and accuracy. Two out of the nine models undergo transfer learning with ImageNet weights, leading to an increase in testing accuracy by 77 percent for VGGNet16 on large dataset. Further, hyperparameter tuning revealed that a lower learning rate of 10-4 and batch size of 128 is best suited for VGG16 and AlexNet. The performance of models with and without data augmentation is analysed, with the generation of T-SNE plots to understand dataset complexity. Overall, the performance of ResNet18 was the best across all three datasets. This report provides a detailed explanation for the observed results.

B. Introduction

Food forms the basis of human existence. In recent years, the demand for food has increased drastically due to population growth, necessitating changes in agricultural methodologies. However, even now, a large number of tasks primarily related to produce picking, sorting and sales are done manually. This consumes a large amount of labour and is also very inefficient. Automated vegetable and fruit classification systems can help farmers in such a scenario. Additionally, classification can be used in inventory management sector to identify and restock produce correctly. It might also benefit kids and the blind as well as enhance grocery store self-checkouts.However, this classification problem is comparatively more challenging because of the produces' wide variety in terms of shape, color, and texture. In some cases, similar-looking produce may rely solely on texture for differentiation, necessitating a deep learning model capable of fine-grained classifications. Factors such as ripeness differences, can contribute to misclassification. Conversely, intra-class samples may contain too many similar images, potentially decreasing the generalisability of the model. One of the existing solutions to improve intra-class diversity is data augmentation in terms of rotations, scaling and color jittering. Further, transfer learning provides a standardised initial weights, resulting in model capable of identifying produce that look similar due to more fine-tuned weights. However, color jittering and excessive shape transformations may over distort the appearance of produce resulting in more misclassifications. Also, transfer learning may also lead to overfitting as the model may over fine-tune its weights. In order to classify these images effectively, we have taken a systematic approach involving several key stages:

1. Dataset Partitioning and models: Initially, we gather three datasets of produce images that vary in number of classes, small, medium and large, ensuring a variability in angles. The datasets undergo transformations to enhance intra-class diversity and address class imbalance through data augmentation. CNN architectures such as, VGGNet16, AlexNet and ResNet18, evaluated through accuracy, precision, recall, and F1-score. Image samples are split into training, validation, and test sets in the ratio 80:10:10 for medium and small, while the large dataset was already split into train (70), validation (15) and test (15) folders.



2. Initial Hyperparameter Exploration: For the small and medium datasets, we initially experimented with a batch size of 64 and found that a learning rate of 0.01 with the Adam optimizer yielded the best results.

3. Performance Analysis: Upon testing these configurations on the large dataset, we observed varied performance across different CNN architectures. While ResNet performed well, VGG16 and AlexNet exhibited suboptimal results.

4. Adaptive Strategy for Large Dataset: In response to the performance disparity, we explored alternative optimization algorithms. Stochastic Gradient Descent (SGD), known for its momentum parameter, showed promise, particularly with VGG16 on the large dataset.

5. Fair Comparison and Hyperparameter Tuning: To maintain fairness in model comparison, we retained the Adam optimizer with a learning rate of 0.01 for the large dataset across all models. However, extensive hyperparameter tuning was conducted, including experimentation with batch sizes and learning rates, detailed in the ablative study within the results section.Various metrics were considered to evaluate the algorithm's performance:

Accuracy: The proportion of correctly classified samples.

Precision: The ratio of correctly predicted positive observations to the total predicted positives.

Recall: The ratio of correctly predicted positive observations to the actual positives.

F1 Score: The harmonic mean of precision and recall, providing a balance between the two.
Mean Loss: The average loss computed during training or evaluation. Models trained from scratch are expected to have much lower accuracy and higher loss than models trained with transfer learning

B.1. Related Works

In this research [13], they improved the performance of vanilla VGGNet model on vegetable classification by combining the outputs of two fully connected layers that achieved an accuracy of 95.8 percent. They further improved this model by including batch normalisation layers that achieved 96.5 percent accuracy. Study in [14] uses image saliency to detect most important features. They used VGGNet and achieved an accuracy of 95.6 percent. Researchers in [15] used a deep neural network that used stochastic gradient descent that processed 3 million learning iterations to achieve an accuracy of 97.58 percent, however, more than 10 million learning iterations resulted in performance decline. Work in [16] entailed vegetable classification using CNN but focuses on reducing misclassifications when vegetables look similar in terms of colour. In paper [17], used a variety of CNN based models pretrained vs trained from scratch on VGGNet16, MobileNet, InceptionV3 and ResNet to ascertain the best performing model. Overall, they concluded that transfer learning produced the best model with over 99.9 percent accuracy in InceptionV3 and ResNet50. In this paper [18], they used a modified AlexNet model that used ReLU instead of the sigmoid function after the output layer that sped up the training process. They achieved a testing accuracy of 92.1 percent.

C. Methodology

C.1. Datasets

We have used 3 different datasets from kaggle for our study:
1. Vegetable Image Dataset [Large dataset]
2. Fresh and Stale Images of Fruits and Vegetables [Medium dataset]
3. Fresh and Rotten Classification [Small dataset]

The large dataset consists of 21,000 images from 15 different classes. Each class of vegetable has a uniform distribution of 1400 images. The dataset is split into 70 percent for training (15,000 images), 15 percent for validation (3,000) and 15 percent for testing (3,000) purposes. The image resolution for the images in the dataset is 224x224.The original medium dataset contains 14 classes and is downsized to 6. There are a total of 7038 images among 6 classes.The small dataset contains 14 classes but we reduced it to 2 classes. There are a total of 3486 images among 2 chosen classes. The dataset is divided into training (2787 samples), testing (348 samples) and validation set (349 samples) in the ratio 80:10:10. All three datasets have varied class representation with rotated images, different illuminations, cut and uncut produce, produce found in different background setting such as market place, kitchen, cutting board, basket and a person holding it. There are also variation in illuminations and the number of produce in a single image for example, some images have a single vegetable whereas other images have at least 7-8 vegetables stacked together. Further, different variants of a produce such as apple (green and red apples) and tomato are included in the dataset to improve the robustness of the trained CNN model.

(a) Bean (b) Brinjal (c) Capsicum

Figure 1. Vegetables of Large Dataset

(a) fresh apple (b) fresh bitter gourd (c) fresh capsicum

Figure 2. Vegetables of Medium Dataset

(a) fresh banana (b) freshokra

Figure 3. Vegetables of Small Dataset

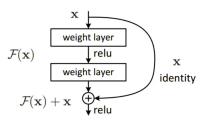
All the three datasets are preprocessed using PyTorch transforms, resizing them to 224x224, flipping horizontally and vertically, and rotating by 10 degrees. Random erasing and colour jittering are applied to introduce variability in colour distribution, aiding the model's ability to recognize vegetables under various lighting conditions. These transformations are combined using 'transforms.compose()' to enhance dataset diversity and mitigate overfitting. Additionally, the medium and small datasets, due to class imbalance, undergo data augmentation using PyTorch's WeightedRandomSampler. First, class weights are calculated inversely proportional to the number of samples in each class.

These weights are then converted to a Double Tensor and passed to WeightedRandomSampler, which returns a class sampler. This sampler facilitates the sampling of data points during training, ensuring that the probability of selecting each sample is proportional to its assigned weight enabling a more balanced representation of classes during the training process. The sampler and training set is then passed as an argument to the DataLoader class to initialise the data loaders for the training set. The Testing set and Validation set do not undergo class balancing. The batch size chosen across all three datasets and three models is 64 initially.

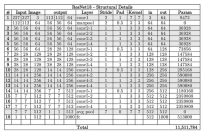
C.2. CNN Models

We have chosen ResNet-18, VggNet-16 and AlexNet for the vegetable classification with their standard layers. Other models like Inception-v3 and DenseNet have high computational complexity compared to ResNet and Vgg which could raise the problem of overfitting on the vegetable dataset. Lighter models like MobileNet might not generalise the images accurately given its trade off between accuracy and computational efficiency. Therefore, considering the complexity of our datasets, the three chosen models seem fit to the vegetable classification task.

1. ResNet18: ResNet18 has 18 layers, with a 7x7 kernel in the first layer, followed by four sets of Convolutional layers with two residual blocks. Every block is composed of two weight layers, and a skip connection that links the output of the second weight layer to the input using a ReLU activation function. The model accepts inputs of size 224x224. A deep model are more likely to capture complex features resulting in a more robust and fine-tuned model. This is imperative to our domain as produce images have high inter-class similarity in terms of color and shape. Skip connections also mitigates vanishing gradient problem, thus resulting in higher performance with an increase in model complexity or number of hidden layers. Additionally, despite ResNet18 having fewer parameters, it results in good performance and better training speeds due to residual connections. ResNet18 has 1.8 Billion flop calculations.



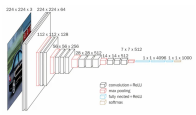
(a) Block Diagram



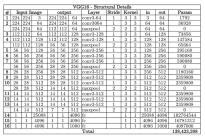
(b) Structural Details

Figure 4. ResNet-18

2. VGGNet16: VGGNet16 has 16 layers and has 13 convolutional layers, 5 max pooling layers, and 2 dense layers. It operates on input of size 224x224 with 3 RGB channels, The convolutional layers use a 3x3 kernel with a stride of 1 and padding of 1 and max-pooling layers have a 2x2 kernel with a stride of 2. The model has similar depth to ResNet18 but focuses on simplicity and does not have skip connections. We selected VggNet to assess whether simplicity in architecture yields comparable results. VGGNET16 has 15.3 billion flop calculations.



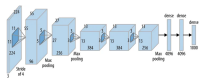
(a) Block Diagram



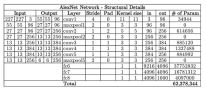
(b) Structural Details

Figure 5. VGGNet-16

3. AlexNet: AlexNet is a light weight model with just eight layers, made up of convolutional layers, max-pooling layers and 3 fully connected layers. It uses ReLU activation function that enables faster convergence of minimum loss during model training. The model also has local response normalisation layers that enhances its generalisability by normalising the outputs across adjacent layers. We opted for AlexNet to ascertain whether a less complex model is suitable for our dataset. AlexNet has 714.2 million flop calculations.



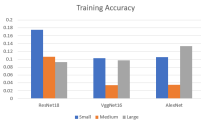
(a) Block Diagram



(b) Structural Details

Figure 6. AlexNet

To analyse computation complexities, all three CNN models were trained on three datasets. Trends on medium, large and small datasets were similar.



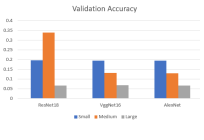


Figure 7. Training (left) and Validation (right) accuracy for one-epoch training

This suggests that the ResNet18's initial weights and architecture are better suited for the problem statement. This can be attributed to deeper layers of ResNet18 and overall, more complex architecture than AlexNet and VggNet16. However, this does not imply that the model will produce the best validation accuracy at the end of model training

phase, but in our case ResNet18 did produce the best results compared to the other two models. From table 1, ResNet18 required the most time for one epoch training for all datasets except medium though less learnable parameters for a model that is 18 layers deep. This may be due to skip connections requiring more resources and computation that has lead to an increase in training time.

Table 1. Wall Clock time for one-epoch training

Dataset	ResNet18	VGGNet16	AlexNet
Small	152.09s	55.094s	45.104s
Medium	65.40s	166.82s	144.160s
Large	1881.07s	65.67s	36.98s

C.3. Optimization Algorithm

To optimize the CNN models for the vegetable and fruit classification task, we employed a systematic approach that involved both validation and optimization. All CNN architectures were evaluated using metrics such as accuracy, precision, recall, and F1-score. Through this validation and optimization process, we aimed to identify the most effective optimization algorithm and hyperparameter configurations for each dataset size and CNN architecture combination, ensuring robust performance across various scenarios. Results are documented in section D.

For the small and medium datasets, as well as for consistency in comparison, the Adam optimizer with a learning rate of 0.01 was selected. Adam is preferred for its adaptive learning rate capabilities, which adjust the learning rates individually for each parameter based on their gradients' first and second moments. We have also experimented with Stochastic gradient descent (SGD) for VGG.

Properties and Hyperparameters:

Optimiser	Adam	SGD
Properties	Adaptive learning rate, momentum, bias correction	Learning rate (lr), momentum, weight decay
Hyperparameters	Learning rate (lr), beta parameters (betas), epsilon (eps)	Learning rate (lr), beta parameters (betas), epsilon (eps)

D. Results

D.1. Experiment Setup

Various techniques were used to improve the vegetable and fruits image classification problem, including resizing images, normalizing pixel values, and enhancing data through rotations, flips, and brightness adjustments. Random weighted sampling was found to be the most effective for most models. However, a batch size of 128 resulted in poorer test accuracy for AlexNet and ResNet18 for medium and small datasets. The small dataset showed a validation accuracy of 99.43 percent, while the medium dataset showed a validation accuracy of 92.89 percent. VggNet16 did not complete 10 epochs, resulting in a low test accuracy of 22.19 percent. For Large Dataset, the initial training was done using the same parameters as was used in small and medium dataset for a fair comparison.

D.2. Main Results

Small dataset: AlexNet and VGGNet16 have a validation and testing accuracy of 18.34 percent, 18.97 percent and 76.85 percent 80.97 percent, respectively. This accuracy dramatically increases to 99.85 percent and 98.43 percent for the validation and testing accuracy for the ResNet respectively could be due to the skip connections that handles the vanishing gradient in an effective manner. ResNet's 18 layers, more than VGGNet and AlexNet, is one of the other reasons for higher accuracy value as the image samples are complex in nature. We observe that the validation accuracy over 10 epochs for the VGGNet and AlexNet alternate between high and low values. The probable reason is that the learning rate is higher than the optimal learning rate necessary to find the local minima, thus missing the local minima. The other reason could be that batch normalisation is not performed, which explains the fluctuation in the accuracies and leading to very large convergence time.

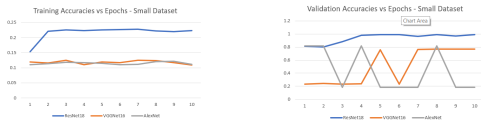


Figure 8. Training (on the left) and Validation (on the right) accuracy vs Epoch on Small Dataset

Medium dataset: ResNet18 has a validation accuracy of 92.6 percent. Whereas, AlexNet and VggNet16 produced a low validation accuracies.

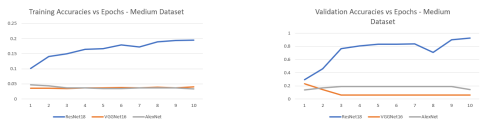


Figure 9. Training (on the left) and Validation (on the right) accuracy vs Epoch on Medium Dataset

This suggests that ResNet18 is able to capture the

most important features in the training samples better than AlexNet and VggNet16. In contrast, VGGNet does not have skip connections, making it more prone to vanishing gradient problems. AlexNet, on the other hand, is a light weight model which may be insufficient to capture the complexity of the training data as there is high intra-class and inter-class similarity.

Large dataset: The validation and the testing accuracies for the VGGNet and AlexNet is low at 6.67 percent for both. ResNet performs the best with the high score of 97.53 percent for the validation accuracy. The low performance of AlexNet and VGGNet stems from the problem of vanishing gradient which is mitigated in the case of ResNet18. The hyperparameters set for the models may not have worked well to accommodate large image sets.

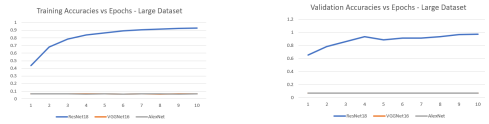


Figure 10. Training (on the left) and Validation (on the right) accuracy vs Epoch on Large Dataset

Table 2. Comparison of 9 models trained from scratch

Sl. No.	Dataset	Model	Accuracy
1	Small	AlexNet	18.97
2	Small	VGG-16	80.97
3	Small	ResNet-18	99.43
4	Medium	AlexNet	15.22
5	Medium	VGG-16	4.5
6	Medium	ResNet-18	92.89
7	Large	AlexNet	6.67
8	Large	VGG-16	6.67
9	Large	ResNet-18	92.49

Transfer Learning: We then proceeded with transfer learning on the large dataset using VGGNet16 and AlexNet as they gave lower accuracy. The model run with VGGNet witnesses a dramatic increase in the test and the validation accuracies with the score of 95.07 percent and 95.6 percent. The likely reason for better results is that the model has better chance of tuning the weights when the weights have already been initialized with the transfer learning. We observe similar results with the AlexNet model, when the learning rate is reduced to 0.001. This tuning shows the increase of the test and the validation accuracies to 84.23 percent and 84.2 percent. We experimented with different learning rates to see the impact, and found that a lower learning rate was better here.

Table 3. Comparison for Transfer Learning - Large Dataset

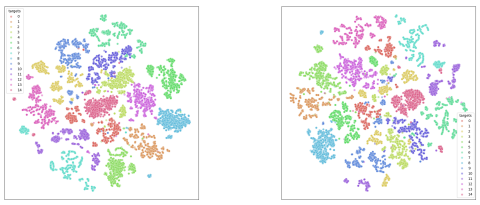
CNN Model	Transfer Learning	Learning Rate	Optimiser	Accuracy
VGGNet	True	0.01	Adam	85.97
AlexNet	True	0.01	Adam	6.67
AlexNet	False	0.001	Adam	76.9
AlexNet	True	0.001	Adam	84.23
VGGNet	False	0.001	SGD	79
VGGNet	True	0.001	SGD	99

Performance of model with class imbalance: We utilised the ResNet18 model on medium dataset to study the impact of data augmentation that uses weighted random sampler to remove class imbalance. ResNet18 produced a testing accuracy of 92.8 percent and 90.75 percent with and without data augmentation respectively. Thus, we can infer that there is a small performance improvement as class weights are inversely proportional to their sample size, which allows the model to learn from the minority classes as effectively as from the majority classes. The recall for minority classes such as fresh bitter gourd and fresh capsicum are only 37.5 percent and 80.3 percent for the imbalanced dataset and 100 percent and 91 percent for balanced dataset.

T-SNE Plots: TSNE plots on medium dataset are plotted after passing image data through the encoder (convolution) layers.

(a) AlexNet (b) ResNet18

Figure 11. TSNE on Medium Dataset



(a) VGGNet16 (b) AlexNet

Figure 12. TSNE with Transfer Learning on Large Dataset

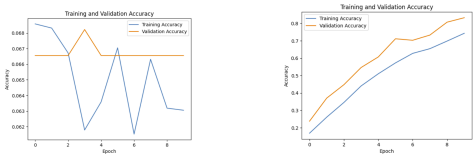
We can observe that the class clusters are more separable and distinct in ResNet18 than in AlexNet. This can be attributed to AlexNet model being less complex than ResNet18 and hence can not extract the high level features that distinguish the different types of produce in the dataset. On the other hand, VGGNet16 and AlexNet models that used pre-trained weights on Large Dataset produce very distinct clusters, better than models that have weights trained from scratch. This is because, transfer learning provides a better initial feature weights. Overall, ResNet18 performs the best in all the 3 cases. However, we observed very low accuracy using AlexNet and VGGNet16 especially for the large dataset. We applied transfer learning with these two models and achieved much better accuracy. For increasing the performance without transfer learning, we experimented with different learning rates and optimisers.

D.3. Ablative Study

Along with just training the 9 models with fixed batch size of 64, learning rate = 0.01, plus 2 transfer learning models, we also performed hyperparameter tuning, particularly using Large Dataset for VGGNet using different learning rates and optimizers. While Adam performs well in terms of optimization, it can lead to worse generalization performance (i.e., poorer performance on unseen data) compared to SGD in image classification tasks when the optimal learning rate and batch size is not provided. With our selected batch size of 64, and 0.01 learning rate, SGD performed better. As SGD performs better than Adam in this context, we can infer from the t-SNE plot that there are tighter or more distinct clustering of data points corresponding to SGD compared to those corresponding to Adam. Tighter clusters suggest that SGD has resulted in features that better separate different vegetable classes in the reduced-dimensional space. Adam's performance might be affected by its inner hyperparameters, which can be computationally expensive to tune.

Table 4. VGG16 - Large Dataset - SGD vs Adam

Optimiser	Learning Rate	Transfer Learning	Accuracy
Adam	0.01	False	6.67
SGD	0.01	False	83.83



(a) Adam (b) SGD

Figure 13. Accuracy Results on Large Dataset

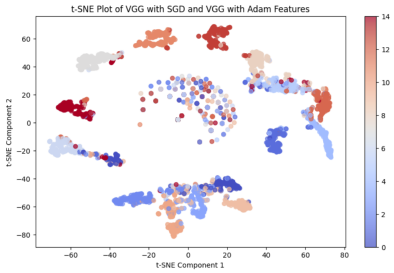
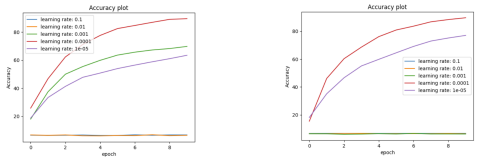


Figure 14. TSNE Results - Adam vs SGD

We also trained AlexNet using different batch sizes and learning rates, as depicted in the graph. We observed that for the large dataset, a batch size of 128 performs the best, while 8 performs the worst (using a fixed Learning rate of 0.01) in Alexnet, and a learning rate of 0.0001 gives best results as opposed to 0.1 which gives the worst accuracy (with fixed Batch Size of 64). For VGGNet also, similar pattern is observed for 5 different learning rates.



(a) AlexNet (b) VGGNet

Figure 15. Accuracy Plot for different learning rates

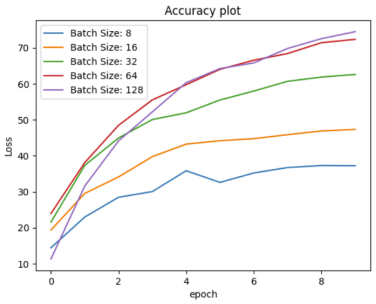


Figure 16. Large dataset - AlexNet - different batch sizes

This is in line with our expectations as larger batch sizes tend to provide more stable gradients during training by reducing the variance in gradient estimates, leading to more consistent weight updates. In case of learning rates, a high learning rate causes large weight updates, which can destabilize training while a very low learning rate leads to small weight updates, slowing down convergence. Hence, tuning is essential, as it helps us identify the best values of the different hyperparameters and see what works best for our model and dataset.

.

References

[1] Ahmed, M. I., Mamun, S. M., Asif, A. U. Z. (2021, May). Dcnn-based vegetable image classification using transfer learning: A comparative study. In 2021 5th International Conference on Computer, Communication and Signal Processing (ICCCSP) (pp. 235-243). IEEE.

[2] Tapia-Mendez, E., Cruz-Albarran, I. A., Tovar-Arriaga, S., Morales-Hernandez, L. A. (2023). Deep Learning-Based Method for Classification and Ripeness Assessment of Fruits and Vegetables. Applied Sciences, 13(22), 12504.

[3] Li, Z., Li, F., Zhu, L., Yue, J. (2020). Vegetable recognition and classification based on improved VGG deep learning network model. International Journal of Computational Intelligence Systems, 13(1), 559-564.

[4] Hameed, K., Chai, D., Rassau, A. (2018). A comprehensive review of fruit and vegetable classification techniques. Image and Vision Computing, 80, 24-44.

[5] Singh, H., Singh, R., Goel, P., Singh, A., Sharma, N. (2022). Automatic Framework for Vegetable Classification using Transfer-Learning. Int. J. Electr. Electron. Res, 10(2), 405-410.

[6] Vegetable Image Dataset: https://www.kaggle.com/datasets/misrakahmed/vegetable-image-dataset

[7] Fresh and Stale Images of Fruits and Vegetables Dataset: https://www.kaggle.com/datasets/raghavrpotdar/fresh-and-stale-images-of-fruits-and-vegetables

[8] Fresh and Rotten Classification Dataset https://www.kaggle.com/datasets/swoyam2609/fresh-and-stale-classification

[9] CNN and Transfer learning: Vegetable Image Classification https://medium.com/@rodrigonev98/cnn-and-transfer-learning-vegetable-image-classification-ecdc1da1fff

[10] Base on Resnet https://www.kaggle.com/code/yuchuanwang/base-on-resnet-99-93-accuracy

[11] Vegetable Classifier - VGG https://www.kaggle.com/code/ashwinshetgaonkar/vegetable-classifier-vgg-pytorch

[12] Difference between AlexNet, VGGNet, ResNet, and Inception https://towardsdatascience.com/the-w3h-of-alexnet-vggnet-resnet-and-inception-7baaaecccc96

[13] Li, Z., Li, F., Zhu, L., Yue, J. (2020). Vegetable recognition and classification based on improved VGG deep learning network model. International Journal of Computational Intelligence Systems, 13(1), 559-564.

[14] Zeng, G. (2017, October). Fruit and vegetables classification system using image saliency and convolutional neural network. In 2017 IEEE 3rd Information Technology and Mechatronics Engineering Conference (ITOEC) (pp. 613-617). IEEE.

[15] Sakai, Y., Oda, T., Ikeda, M., Barolli, L. (2016, July). A vegetable category recognition system using deep neural network. In 2016 10th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS) (pp. 189-192). IEEE.

[16] Duth, P. S., Jayasimha, K. (2020, May). Intra class vegetable recognition system using deep learning. In 2020 4th International conference on intelligent computing and control systems (ICICCS) (pp. 602-606). IEEE.

[17] Ahmed, M. I., Mamun, S. M., Asif, A. U. Z. (2021, May). DCNN-based vegetable image classification using transfer learning: A comparative study. In 2021 5th International Conference on Computer, Communication and Signal Processing (ICCCSP) (pp. 235-243). IEEE.

[18] Zhu, L., Li, Z., Li, C., Wu, J., Yue, J. (2018). High performance vegetable classification from images based on alexnet deep learning model. International Journal of Agricultural and Biological Engineering, 11(4), 217-223.

[19] Resnet18 Model With Sequential Layer For Computing Accuracy On Image Classification Dataset https://ijcrt.org/papers/IJCRT2205235.pdf

[20] Everything you need to know about VGG16 https://medium.com/@mygreatlearning/everything-you-need-to-know-about-vgg16-7315defb5918

E. Supplementary Material

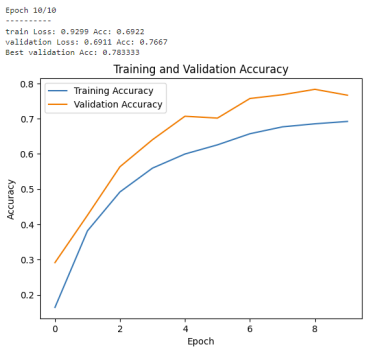


Figure 17. AlexNet - No transfer learning - 0.001 Learning Rate, Batch size 64 - LargeDataset - Train vs Validation accuracy

Test Accuracy: 0.7690
Precision: 0.7870
Recall: 0.7690
F1 Score: 0.7694
Mean Loss: 0.7000

	precision	recall	f1-score	support
Bean	0.50	0.56	0.53	200
Bitter_Gourd	0.69	0.78	0.73	200
Bottle_Gourd	0.96	0.63	0.76	200
Brinjal	0.70	0.69	0.70	200
Broccoli	0.75	0.82	0.78	200
Cabbage	0.76	0.61	0.68	200
Capsicum	0.98	0.81	0.89	200
Carrot	0.94	0.94	0.94	200
Cauliflower	0.89	0.77	0.82	200
Cucumber	0.64	0.74	0.69	200
Papaya	0.78	0.83	0.80	200
Potato	0.79	0.99	0.88	200
Pumpkin	0.66	0.91	0.76	200
Radish	0.85	0.85	0.85	200
Tomato	0.92	0.61	0.74	200
accuracy			0.77	3000
macro avg	0.79	0.77	0.77	3000
weighted avg	0.79	0.77	0.77	3000

Figure 18. AlexNet - No transfer learning - 0.001 Learning Rate, Batch size 64 - LargeDataset - Metrics

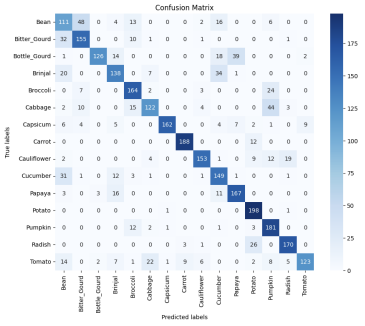


Figure 19. AlexNet - No transfer learning - 0.001 Learning Rate, Batch size 64 - LargeDataset - Confusion Matrix

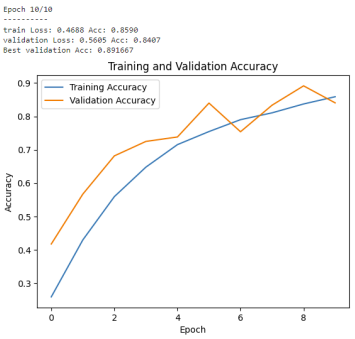


Figure 20. ResNet - No transfer learning - 0.01 Rate, Batch size 64 - LargeDataset - Train vs Validation accuracy

Test Accuracy: 0.8597
Precision: 0.8857
Recall: 0.8597
F1 Score: 0.8608
Mean Loss: 0.5217

	precision	recall	f1-score	support
Bean	0.74	0.68	0.71	200
Bitter_Gourd	0.56	0.95	0.70	200
Bottle_Gourd	0.98	0.85	0.91	200
Brinjal	0.77	0.89	0.83	200
Broccoli	0.81	0.94	0.87	200
Cabbage	0.87	0.86	0.86	200
Capsicum	0.98	0.94	0.96	200
Carrot	0.99	0.97	0.98	200
Cauliflower	0.93	0.85	0.89	200
Cucumber	0.97	0.56	0.72	200
Papaya	0.98	0.63	0.77	200
Potato	0.99	0.98	0.99	200
Pumpkin	0.76	0.98	0.86	200
Radish	1.00	0.93	0.96	200
Tomato	0.94	0.89	0.91	200
accuracy			0.86	3000
macro avg	0.89	0.86	0.86	3000
weighted avg	0.89	0.86	0.86	3000

Figure 21. ResNet - No transfer learning - 0.01 Rate, Batch size 64 - LargeDataset - Metrics

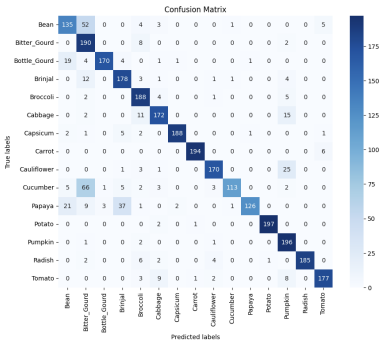


Figure 22. ResNet - No transfer learning - 0.01 Rate, Batch size 64 - LargeDataset - Confusion Matrix

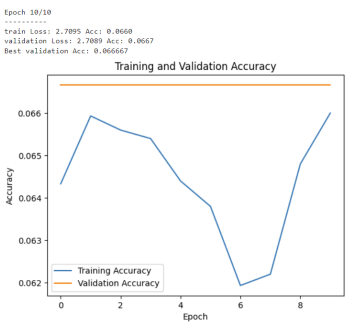


Figure 23. AlexNet - No transfer learning - 0.01 Rate, Batch size 64 - LargeDataset - Train vs Validation accuracy

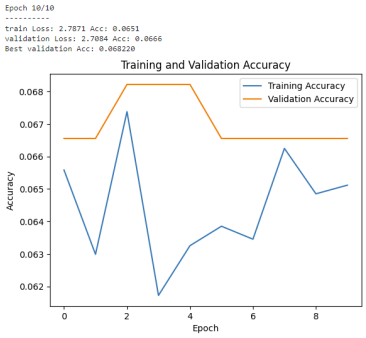
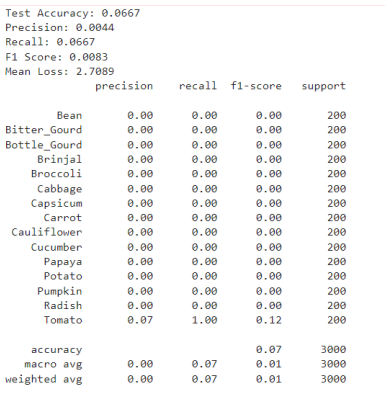


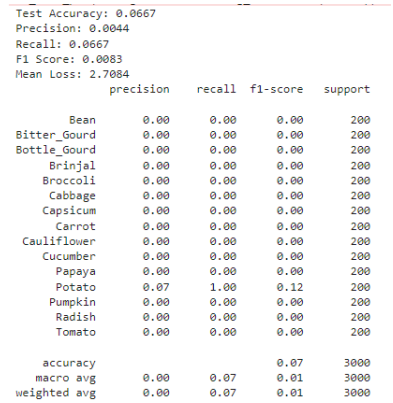
Figure 26. VGGNet - No transfer learning - 0.01 Rate, Batch size 64 - LargeDataset - Train vs Validation accuracy



Test Accuracy: 0.0667
Precision: 0.0044
Recall: 0.0667
F1 Score: 0.0083
Mean Loss: 2.7089

	precision	recall	f1-score	support
Bean	0.00	0.00	0.00	200
Bitter_Gourd	0.00	0.00	0.00	200
Bottle_Gourd	0.00	0.00	0.00	200
Brinjal	0.00	0.00	0.00	200
Broccoli	0.00	0.00	0.00	200
Cabbage	0.00	0.00	0.00	200
Capsicum	0.00	0.00	0.00	200
Carrot	0.00	0.00	0.00	200
Cauliflower	0.00	0.00	0.00	200
Cucumber	0.00	0.00	0.00	200
Papaya	0.00	0.00	0.00	200
Potato	0.00	0.00	0.00	200
Pumpkin	0.00	0.00	0.00	200
Radish	0.00	0.00	0.00	200
Tomato	0.07	1.00	0.12	200
accuracy			0.07	3000
macro avg	0.00	0.07	0.01	3000
weighted avg	0.00	0.07	0.01	3000

Figure 24. AlexNet - No transfer learning - 0.01 Rate, Batch size 64 - LargeDataset - Metrics



Test Accuracy: 0.0667
Precision: 0.0044
Recall: 0.0667
F1 Score: 0.0083
Mean Loss: 2.7084

	precision	recall	f1-score	support
Bean	0.00	0.00	0.00	200
Bitter_Gourd	0.00	0.00	0.00	200
Bottle_Gourd	0.00	0.00	0.00	200
Brinjal	0.00	0.00	0.00	200
Broccoli	0.00	0.00	0.00	200
Cabbage	0.00	0.00	0.00	200
Capsicum	0.00	0.00	0.00	200
Carrot	0.00	0.00	0.00	200
Cauliflower	0.00	0.00	0.00	200
Cucumber	0.00	0.00	0.00	200
Papaya	0.00	0.00	0.00	200
Potato	0.07	1.00	0.12	200
Pumpkin	0.00	0.00	0.00	200
Radish	0.00	0.00	0.00	200
Tomato	0.00	0.00	0.00	200
accuracy			0.07	3000
macro avg	0.00	0.07	0.01	3000
weighted avg	0.00	0.07	0.01	3000

Figure 27. VGGNet - No transfer learning - 0.01 Rate, Batch size 64 - LargeDataset - Metrics

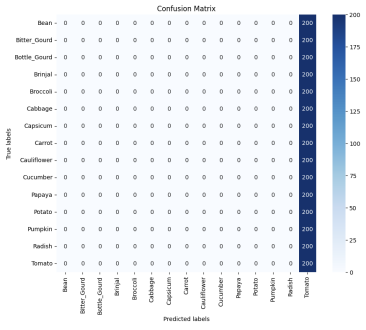


Figure 25. AlexNet - No transfer learning - 0.01 Rate, Batch size 64 - LargeDataset - Confusion Matrix

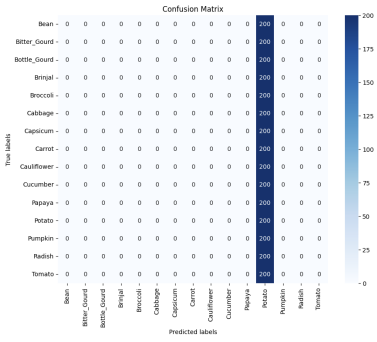


Figure 28. VGGNet - No transfer learning - 0.01 Rate, Batch size 64 - LargeDataset - Confusion Matrix

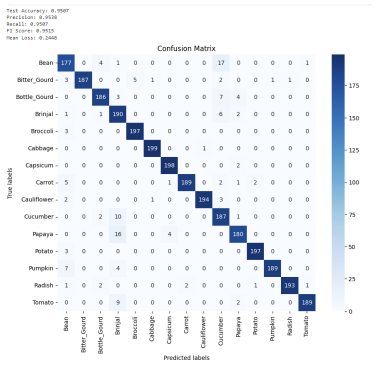


Figure 29. VGGNet - WithTransferLearning - Batch64 - LR0.01 - LargeDataset - Metrics and Confusion Matrix

Figure 30. AlexNet - NoTransferLearning - Batch64 - LR0.01 - Medium - Metrics and Accuracy curve

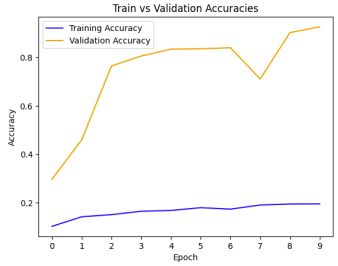


Figure 31. ResNet - NoTransferLearning - Batch64 - LR0.01 - Medium - Train vs Validation accuracy

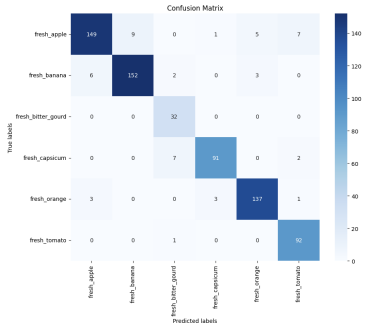
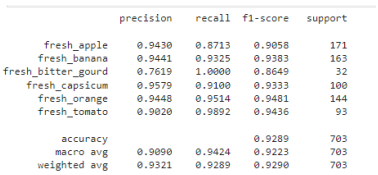


Figure 32. ResNet - NoTransferLearning - Batch64 - LR0.01 - Medium - Metrics



	precision	recall	f1-score	support
fresh_apple	0.9430	0.8713	0.9058	171
fresh_banana	0.9441	0.9325	0.9383	163
fresh_bitter_gourd	0.7619	1.0000	0.8649	32
fresh_capsicum	0.9579	0.9100	0.9333	100
fresh_orange	0.9448	0.9514	0.9481	144
fresh_tomato	0.9020	0.9892	0.9436	93
accuracy			0.9289	703
macro avg	0.9090	0.9424	0.9223	703
weighted avg	0.9321	0.9289	0.9290	703

Figure 33. ResNet - NoTransferLearning - Batch64 - LR0.01 - MediumDataset - Confusion Matrix

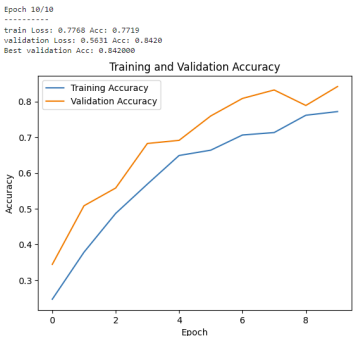


Figure 34. AlexNet - WithTransferLearning - Batch32 - LR0.001 - LargeDataset - Train vs Validation accuracy

Test Accuracy: 0.8423
Precision: 0.8546
Recall: 0.8423
F1 Score: 0.8427
Mean Loss: 0.5686

	precision	recall	f1-score	support
Bean	0.73	0.68	0.70	200
Bitter_Gourd	0.90	0.92	0.91	200
Bottle_Gourd	0.93	0.71	0.80	200
Brinjal	0.85	0.80	0.82	200
Broccoli	0.96	0.85	0.90	200
Cabbage	0.94	0.69	0.80	200
Capsicum	0.99	0.84	0.91	200
Carrot	0.97	0.97	0.97	200
Cauliflower	0.68	0.85	0.76	200
Cucumber	0.75	0.72	0.74	200
Papaya	0.71	0.94	0.80	200
Potato	0.98	0.87	0.92	200
Pumpkin	0.83	0.97	0.90	200
Radish	0.83	0.94	0.88	200
Tomato	0.76	0.88	0.81	200
accuracy			0.84	3000
macro avg	0.85	0.84	0.84	3000
weighted avg	0.85	0.84	0.84	3000

Figure 35. AlexNet - WithTransferLearning - Batch32 - LR0.001 - LargeDataset - Metrics

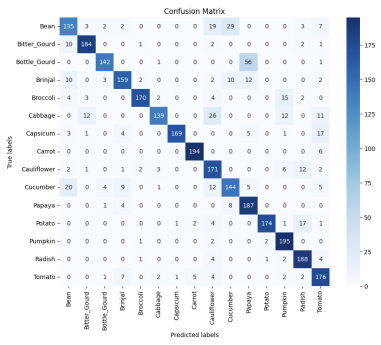
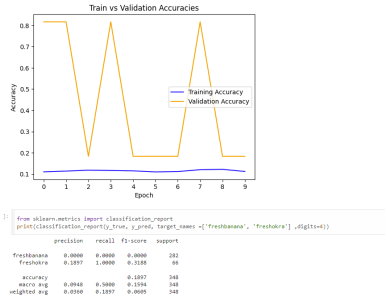


Figure 36. AlexNet - WithTransferLearning - Batch32 - LR0.001 - LargeDataset - Confusion Matrix



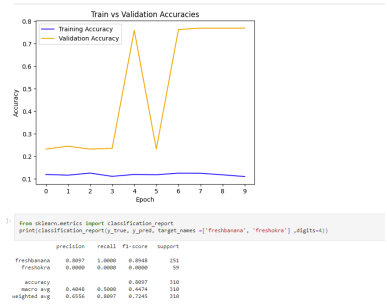

```
from sklearn.metrics import classification_report
print(classification_report(y_true, y_pred, target_names =['freshbanana', 'freshokra'] ,digits=4))
```

```
              precision    recall  f1-score   support

freshbanana     0.0000    0.0000    0.0000       282
  freshokra     0.1897    1.0000    0.3188        66

   accuracy                         0.1897       348
  macro avg     0.0948    0.5000    0.1594       348
weighted avg    0.0360    0.1897    0.0605       348
```

Figure 37. AlexNet - NoTransferLearning - Batch64 - LR0.01 - Small - Train vs Validation accuracy




```
from sklearn.metrics import classification_report
print(classification_report(y_true, y_pred, target_names =['freshbanana', 'freshokra'] ,digits=4))
```

```
              precision    recall  f1-score   support

freshbanana     0.8097    1.0000    0.8948       251
  freshokra     0.0000    0.0000    0.0000        59

   accuracy                         0.8097       310
  macro avg     0.4048    0.5000    0.4474       310
weighted avg    0.6556    0.8097    0.7245       310
```

Figure 38. VGGNet - NoTransferLearning - Batch64 - LR0.01 - Small - Train vs Validation accuracy

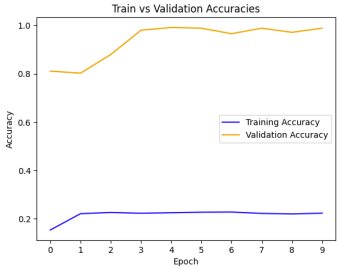


Figure 39. ResNet - NoTransferLearning - Batch64 - LR0.01 - Small - Train vs Validation accuracy

```
              precision    recall  f1-score   support

fresh_banana     0.9929    1.0000    0.9965       281
  fresh_okra     1.0000    0.9701    0.9848        67

    accuracy                         0.9943       348
   macro avg     0.9965    0.9851    0.9907       348
weighted avg     0.9943    0.9943    0.9942       348
```

Figure 40. ResNet - NoTransferLearning - Batch64 - LR0.01 - Small - Metrics

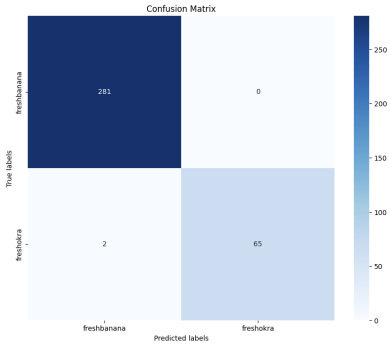


Figure 41. ResNet - NoTransferLearning - Batch64 - LR0.01 - Small - Confusion Matrix