Dry Bean Classification Using Deep Learning

Urvashi Dube, Student at Northeastern University, Vancouver, dube.u@northeastern.edu

Sudhish Subramaniam, Student at Northeastern University, Vancouver, subramaniam.su@northeastern.edu

Abstract

The accurate classification of diverse dry bean varieties based on visual features and characteristics has been a significant challenge for classification algorithms in recent times. This research project aims to address this issue by proposing a highly accurate neural network system capable of classifying fourteen distinct dry bean varieties. Leveraging the power of deep learning techniques, including Feedforward Neural Networks, Convolutional Neural Networks, maximum and average pooling, batch normalization, and dense layers, we process a dataset comprising 33,000 images of the bean varieties in the introduced model. By meticulously tuning hyperparameters and incorporating multiple layers in our models, we achieve a remarkable maximum accuracy of 90.79%. Extensive model testing on the dataset and comprehensive performance evaluations are conducted to determine the optimal classifier, facilitating a detailed model comparison. The successful implementation of this system promises significant advancements in efficiently and precisely identifying bean varieties, thereby benefiting the agricultural industry through improved quality control, market segmentation, and informed decision-making for pest management and breeding endeavours.

Keywords:

1. Introduction

1.1. Overview

In recent years, achieving high accuracy in dry bean classification has been a persistent challenge. To address this issue, researchers have obtained a dataset of 33,000 precisely captured images of dry beans within a controlled environment, eliminating background noise. With fourteen distinct bean varieties in the dataset, deep learning methods are employed to develop a robust classification model for accurate bean categorization. This model holds significant potential to benefit farmers and industries, ensuring precise dry bean classification, which in turn enhances product quality and aids in market segmentation. Additionally, the model's ability to assess each bean's quality empowers farmers to make informed decisions regarding pesticide usage for optimal production outcomes.

1.2. Motivation

The accurate classification of diverse dry bean varieties using advanced neural network systems is crucial for the agricultural community. This research holds significant implications for enhancing agricultural efficiency, quality control, and market segmentation. Precise identification enables farmers to optimize resource usage, leading to improved crop yields, while ensuring consumer trust and satisfaction through targeted market segmentation. Moreover, accurate classification fosters sustainable practices by facilitating targeted pest management and disease resistance measures, reducing reliance on broad-spectrum pesticides. Understanding the genetic diversity of bean varieties also aids in developing resilient cultivars with enhanced traits. Overall, this research demonstrates the transformative potential of deep learning methods in revolutionizing agriculture and contributing to global food security and sustainability.

1.3. Approach

In this study, we processed a dataset of 33,000 images using various tools and applied different convolutional neural networks (CNNs). We experimented with different pre-trained weights, including imagenet, ResNet, VGG16, and VGG19 and own CNN network as well as random weights to address the classification problem. By extensively tuning hyperparameters and comparing the accuracy of all models, we identified the most suitable model for classification and predictions. The use of diverse CNN combinations resulted in significantly improved accuracy compared to traditional machine learning algorithms, making this method the optimal choice for classification tasks.

1.4. Dataset Used

The dataset used in this research contains 33,064 images representing 14 distinct dry bean varieties. These images were captured meticulously using a specialized illumination box to ensure the elimination of shadows and background noises. Furthermore, the images were converted into grayscale for streamlined processing and analysis. The dataset can be accessed at http://www.muratkoklu.com/datasets/Dry_Bean_Image_Dataset.zip.

2. Background

The papers till date demonstrate a series of studies and research on classifying dry beans using computer vision and machine learning techniques. The main goal is to develop automatic methods for identifying and classifying different varieties of dry beans based on their features and characteristics. [1] introduces a computer vision system for distinguishing seven different registered varieties of dry beans based on 16 extracted features. They compared the performance of several classification models, including MLP, SVM, kNN, and DT, achieving overall correct classification rates above 87%. [2] had developed a computer vision system to inspect beans' quality based on size and color quantification. The system achieved high accuracy in classifying different bean types using artificial neural networks. [3] explores a deep neural network-based approach to automatically classify seven varieties of dry beans, achieving an accuracy of 93.44% and an F1-score of 94.57%. The deep neural network outperformed traditional machine learning approaches in this task. [4] focuses on classifying dry beans based on their dimensions and size using a dataset of 13,611 grains images. They achieved an impressive classification accuracy of 99.5% with their proposed technique. [5] utilizes 22 machine learning algorithms to classify seven varieties of dry beans. The CatBoost ML algorithm achieved the highest overall mean accuracy of 93.8%, outperforming other techniques. [6] explores a hybrid structure of extreme learning machine (ELM) and GoogLeNet transfer learning with the salp swarm algorithm (SSA) for classifying 14 different types of dry beans. The proposed model achieved a success rate of 91.43%, outperforming traditional machine learning algorithms. [7] analyzes a dataset of over 13k samples of dry beans using machine learning and deep learning techniques. Various classifiers like Multinomial Bayes, Support Vector Machines, Decision Tree, Random Forest, Voting Classifier, and Artificial Neural Network were used, obtaining overall accuracies ranging from 88.35% to 93.61%. Overall, these studies demonstrate the effectiveness of using computer vision and machine learning techniques for dry bean classification. They showcase different approaches and algorithms, achieving high accuracy rates in distinguishing various bean varieties. The research lays a solid foundation for the development of practical applications for seed quality control and sustainable agricultural systems.

3. Approach

Preprocessing is essential for our dataset containing 33,000 images across 14 different categories. However, some categories have images with varying dimensions, such as 600*600 pixels. This variation could impact our model's performance. To address this, we gather all the images into a single folder and utilize a Python code to resize them uniformly to 256*256 pixels, as depicted in the fig. 1 and fig. 2.

After transforming the images, we import crucial libraries required for our model. These include TensorFlow, as well as various Keras layers such as Input, Dense, Flatten, and BatchNormalization. Additionally, we import functions like save_model and plot_model from tensorflow.keras.models and tensorflow.keras.utils respectively. Moreover, we make use of the resnet50 module from tensorflow.keras.applications for its functionality. To avoid any further warnings during the execution of our code, we also import the warnings library.

Fig. 1: Battal (Dry Bean)

Fig. 2: Zirve (Dry Bean)

The entire model is designed to process images of a fixed size, specifically 256*256 pixels. During the training process, we set the batch size for our neural network to 32, meaning it will process 32 images in each training iteration.

To load the dataset containing 33,000 images, we utilize tf.keras.preprocessing.image_dataset_from_directory library. This function helps us organize the data into separate training and validation datasets. The training dataset consists of 80% of the images, while the remaining 20% is used for validation purposes. This division ensures that we have sufficient data to train the model effectively while also having a separate set to evaluate its performance.

```
In [3]: print(class_names)
['Battal', 'Beysehir', 'Big_Dermason', 'Bombay', 'Cali', 'Dermason', 'Hinis', 'Horoz',
'Nirvana', 'Ozmen', 'Seker', 'Sira', 'Ustun42', 'Zirve']
```

Fig. 3: 14 Categories of dry beans

The 14 classes of dry beans in the training dataset are clearly listed and displayed in the fig. 3. Once the train and validation datasets are generated, the labels play a crucial role in the neural network model. To handle the labels, we employ one hot encoding, which sets up appropriate labels for both the training and validation data.

In the process of creating the model, we first define the input layer with an input shape of 256*256 pixels. Next, we import the base layer, which is the ResNet50 model. Along with importing the ResNet model, we also bring in the pre-trained weights from the ImageNet dataset. By doing this, we benefit from the knowledge gained during pre-training on ImageNet.

To preserve the knowledge in the pre-trained weights, we freeze them and then proceed to add additional layers to enhance the model's accuracy. The layers we add include Flatten, Dense_1, and BatchNorm_1, all using the "ReLU" activation function as depicted in the fig.4. Finally, we incorporate the output layer with the "SoftMax" activation function (shown in fig. 5) to provide the model's final predictions.

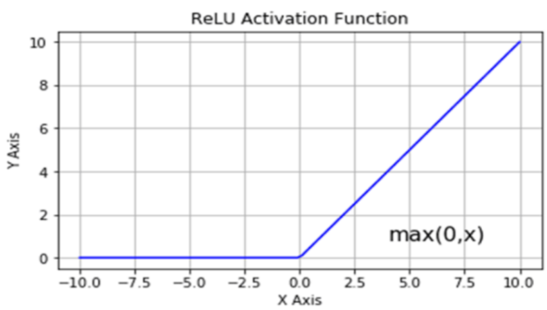

Fig. 4: RelU Activation Function

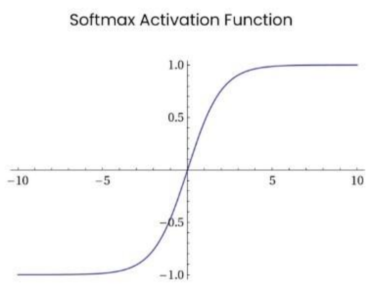

Fig. 5: Softmax Activation Function

After creating the model using `tf.keras.models.Model` to integrate the input and output layers, we print the model summary.In general the model consists of 5 convolutional blocks with flatten, dense (Relu), batch normalization and dense(softmax) layers at the last. Each convolutional layer consists of different combinations of convolutions, batch normalization, relu activation and add block. Once the model is displayed, it is compiled using the Adam optimizer with a learning rate of 1e-3, categorical cross-entropy as the loss function, and accuracy as the evaluation metric.

The model is then trained using the training dataset and validated with the validation dataset for a total of 5 epochs, with each epoch comprising 350 steps. These figures were determined based on the number of images and the laptop's RAM capacity to manage memory usage and time complexity. As a result, only 14,000 images are processed during the training.

To achieve the model with the highest accuracy, various permutations of convolutional and dense layers were explored and evaluated. Different models such as Xception, ResNet50, VGG-19, VGG-16, a custom-built model, a custom model initialized with Inception weights, a custom model with ResNet50 weights, and a custom model with VGG16 weights were applied to the image dataset. The results obtained from these models will be further discussed and analysed in the study.

4. Results

As shown in the previous fig. 1 and fig. 2, the 14 different dry beans are black and white images with no background noise. 14000 images are used that is 1000 images of each category to decrease the time complexity and keeping the laptop’s ram capacity in mind. Different experiments such applying the pretrained weights and algorithms and then modifying these algorithms by adding some convolutional layers is done. Pre trained algorithms applied are VGG-16. VGG-19, resnet50, Xception and three other own models where resenet50, inception models are modified by adding convolution and one model with own number convolutions and own weights are used. Hyperparameter tuning and with different epochs and different steps per epoch are used in the models. Different number of convolutional layers, flatten, batch normalization and dense layers are used.

In the initial stage, when we used 5 number of epochs to train our dataset, we got the results shown in Table 1.

Model	Accuracy
Resnet 50	78.9
VGG19	62.19
VGG16	61.25
Xception	60.16
Resnet Modified	58.38
Inception	53.3

Table 1: Accuracy of pre-trained models on dataset

Keeping these results in mind we take into consideration all these models to increase the number of epochs and steps per epochs.

We then increase the number of epochs by increasing the number of steps per epoch to 350. Hence we get the results shown in Table 2.

Training with 5 epochs and 350 steps per epoch		
Sr. no.	**Model**	**Accuracy**
1	Model 4 (Resnet50 Weights)	90.79
2	Own Model	88.3
3	VGG-19	86.02
4	VGG-16	83.95
5	Resnet50	80.16
6	Model 3 (Inception Weights)	75.29
7	Model 2 (Without Pretrained Weights)	67.31
8	Xception	64.18

Table 2: Accuracy of own models after increasing epochs and steps per epoch

Here the own model is developed using VGG16 Weights + Flatten + Dense + BatchNorm + Dense + BatchNorm + Dense layers.

On fitting all the models in the data we have found that total parameters, trainable parameters and non-trainable parameters play a very important role in model selection. Hence, we have listed down the parameters used in each model in Table 3.

Sr. No.	Models	Total Params	Trainable Params	Non - Trainable Params
1	Model 4	57147022	33558798	23588224
2	Model 3	40713646	18910094	21803552
3	Resnet50	24643982	1056270	23587712
4	Model 2	23139790	23139022	768
5	Xception	22696502	18535022	20861480
6	VGG-19	20483150	458766	20024384
7	VGG-16	15173454	458766	14714688
8	Own Model	2319790	8424334	14715456

Table 3: Total Parameters, trainable parameters and non-trainable parameters of all own models after increasing epochs

Here Total params are the total number of values (weights and biases) in the neural network, trainable params are the number of values that the network learns and updates during training to make predictions better and non-trainable params are the number of fixed values used for other purposes like normalization or preprocessing and don't change during training.

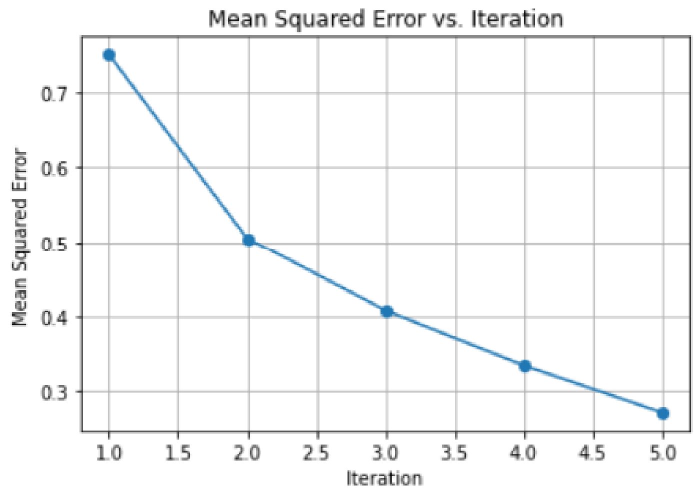

Fig. 6: Graph of Mean Squared Error vs Iterations

Fig. 6 clearly highlights that the mean squared error decreases with the increase in number of epochs.

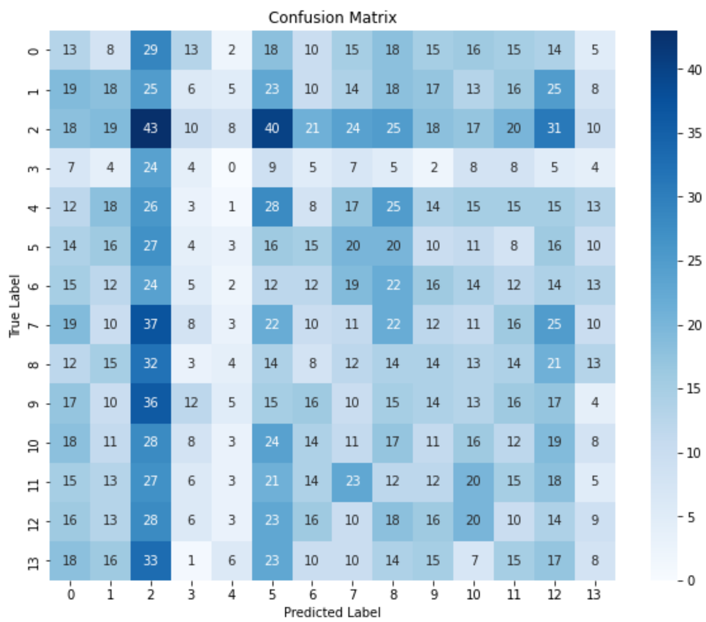

Fig. 7: Confusion Matrix of Developed Model

Model 4 gives an accuracy of 90.79% and shows a confusion matrix in Fig.7. This figure clearly shows that dry bean 0, 1, 2, 5, 6, 7, 8, 9, 10 are correctly predicted. More data has to be incorporated to increase the accuracy of the prediction of the model.

5. Discussion

The research results indicate that Model 4, utilizing the Resnet50 model with pretrained weights from imagenet, along with flatten, dense, batch_ normalization, and dense layers, outperforms all other models in the study. This model has the highest number of total parameters, with a count of 57,147,022.

Additionally, it achieves an impressive maximum accuracy of 90.79% on the given task. The detailed performance metrics, including loss, accuracy, validation loss, and validation accuracy across different epochs, are graphically presented in the Fig. 8.

```
==============================================================================================
Total params: 57147022 (218.00 MB)
Trainable params: 33558798 (128.02 MB)
Non-trainable params: 23588224 (89.98 MB)
______________________________________________________________________________________________
Epoch 1/5
350/350 [==============================] - 786s 2s/step - loss: 0.7517 - accuracy: 0.7338 - val_loss: 0.6885 -
val_accuracy: 0.7457
Epoch 2/5
350/350 [==============================] - 850s 2s/step - loss: 0.5034 - accuracy: 0.8163 - val_loss: 0.5686 -
val_accuracy: 0.7989
Epoch 3/5
350/350 [==============================] - 846s 2s/step - loss: 0.4047 - accuracy: 0.8518 - val_loss: 0.5787 -
val_accuracy: 0.7964
Epoch 4/5
350/350 [==============================] - 786s 2s/step - loss: 0.3244 - accuracy: 0.8829 - val_loss: 0.6668 -
val_accuracy: 0.7821
Epoch 5/5
350/350 [==============================] - 891s 3s/step - loss: 0.2653 - accuracy: 0.9079 - val_loss: 0.6453 -
val_accuracy: 0.7929
C:\Users\urvas\AppData\Local\Temp\ipykernel_6292\1921603258.py:117: UserWarning: You are saving your model as an HDF5 file
via `model.save()`. This file format is considered legacy. We recommend using instead the native Keras format, e.g.
`model.save('my_model.keras')`.
  save_model(model, 'C:/Users/urvas/Projects/Dry Beans Classification Projects/Programs Used/Models/My_Model_4.h5');
```

Fig. 8: Epochs, Accuracy (Train and Validation), loss of Developed model

The findings suggest that incorporating a complex architecture, such as Resnet50, along with pretrained weights from imagenet, can substantially improve the model's performance in the specific task. The inclusion of flatten, dense, batch_normalization, and dense layers seems to have contributed significantly to the model's ability to learn intricate patterns from the data, leading to its superior performance over alternative architectures.

In a broader context, these results highlight the importance of leveraging transfer learning through pretrained models for achieving better performance in deep learning tasks. The success of Model 4, with its high accuracy and complex architecture, suggests that exploring and adapting advanced pretrained models can lead to significant advancements in various real-world applications, particularly in image recognition tasks.

Moving forward, this study identifies several promising avenues to enhance the performance of the model and extend the implications of the research. Firstly, the utilization of data augmentation techniques emerges as a potential strategy to diversify and expand the training dataset, ultimately improving the model's ability to generalize to new and unseen examples. Secondly, given the substantial number of parameters in Model 4, exploring regularization approaches, such as L1/L2 regularization or dropout, presents an opportunity to address overfitting issues and enhance the model's generalization capacity to unseen data. Thirdly, investigating ensemble methods, which involve combining predictions from multiple models, offers a potential means of enhancing the overall performance and robustness of the model.

Additionally, a comprehensive hyperparameter optimization process could reveal optimal configurations that enhance accuracy and convergence speed. Furthermore, exploring alternative architectures beyond Resnet50 and analyzing their impact on performance can provide valuable insights for selecting models tailored to specific tasks. The evaluation of Model 4 with pretrained weights from diverse sources or domain-specific pretrained models may uncover variations in results and suitability for specific datasets. Lastly, delving into interpretability techniques can deepen the understanding of the model's decision-making process and identify critical features contributing to predictions, thereby increasing trustworthiness and practical applicability in real-world scenarios.

In conclusion, the findings underscore the significance of leveraging pretrained models and complex architectures to enhance accuracy in deep learning tasks, offering a robust foundation for future research endeavours aimed at advancing the field of deep learning, particularly concerning image recognition and its diverse applications. Enhanced hardware with greater RAM capacity could be utilized in the future to train models using more images and data augmentations, resulting in reduced time complexity and potentially leading to increased accuracy.

6. Conclusion

In this project, we aimed to improve the image recognition performance for a dataset containing 33,000 images of dry beans across 14 categories. The variations in image dimensions posed a challenge, which we addressed by resizing all images to a uniform size of 256*256 pixels. Leveraging the power of pre-trained ResNet50 model with ImageNet weights, we designed a model with additional layers to enhance accuracy. One hot encoding was used to handle labels, and the model was trained on 80% of the dataset, with the remaining 20% used for validation.

The main discoveries from our project can be summarized as follows. Firstly, preprocessing steps such as image resizing, and one hot encoding play a crucial role in enhancing the model's performance and effectively managing varying image dimensions. Secondly, leveraging the knowledge acquired from pre-trained weights on ImageNet significantly improves the accuracy of our model in the specific task of image recognition. Lastly, the incorporation of ResNet50 along with additional layers, namely Dense_1 and

BatchNorm_1, results in substantial advancements in the accuracy of our image recognition system. These findings underscore the importance of thoughtful preprocessing and the benefits of leveraging pre-trained weights and complex architectures in achieving superior image recognition performance.

The takeaway message of our research is that careful preprocessing and leveraging pre-trained weights are essential for achieving high accuracy in image recognition tasks. The utilization of well-known models like ResNet50, coupled with additional layers, can significantly boost the model's performance. These findings hold valuable implications for the field of deep learning and image recognition applications.

7. Acknowledgements

I would like to extend my sincere gratitude to Professor Jerome Braun for his invaluable support and guidance throughout this research project on "Enhance Dry Bean classification using deep learning." His expertise in the field of deep learning has been instrumental in shaping the direction of this study. I am also thankful to the team behind the website https://www.muratkoklu.com/datasets/ for providing the dataset, which proved to be a crucial resource in training and validating our deep learning model. Their contribution significantly enhanced the quality and relevance of our research. Without their assistance, this work would not have been possible.

8. References

1. Koklu, M., & Ozka, I. A. (2020). Multiclass classification of dry beans using computer vision and machine learning techniques. Elsevier Computers and Electronics in Agriculture, 105507. https://doi.org/10.1016/j.compag.2020.105507
2. Kilic, K., Ismail Hakkı Boyaci, Hamit Köksel and I. Küsmenoĝlu. “A classification system for beans using computer vision system and artificial neural networks.” *Journal of Food Engineering* 78 (2007): 897-904.
3. M. M. Hasan, M. U. Islam and M. J. Sadeq, "A Deep Neural Network for Multi-class Dry Beans Classification," 2021 24th International Conference on Computer and Information Technology (ICCIT), Dhaka, Bangladesh, 2021, pp.1-5,doi: 10.1109/ICCIT54785.2021.9689905.
4. V. Gautam, N. K. Trivedi, A. Anand and J. Rani, "Feature selection based Dry-Beans Multiclass classification with Optimized Deep Neural Network," 2022 10th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) [1](ICRITO), Noida, India, 2022, pp. 1-5, doi: 10.1109/ICRITO56286.2022.9964754.
5. S. Krishnan, S. K. Aruna, Karthick Kanagarathinam, Ellappan Venugopal, "Identification of Dry Bean Varieties Based on Multiple Attributes Using CatBoost Machine Learning Algorithm", Scientific Programming, vol. 2023, Article ID 2556066, 21 pages, 2023. https://doi.org/10.1155/2023/2556066
6. Dogan, M., Taspinar, Y. S., Cinar, I., Kursun, R., Ozkan, I. A., & Koklu, M. (2022). Dry bean cultivars classification using deep CNN features and salp swarm algorithm based extreme learning machine. Computers and Electronics in Agriculture,107575. https://doi.org/10.1016/j.compag.2022.107575
7. Słowiński, G. (Year). Dry Beans Classification Using Machine Learning. University of Technology and Economics, ul. Jagiellońska 82f, 03-301 Warsaw, Poland.
8. Dataset Link: https://www.muratkoklu.com/datasets/

Urvashi Dube (UD) & Sudhish Subramaniam (SS), both contributed equally. Both found the dataset, researched on the topic, designed models, performed experiments, ran all models, developed research paper and presentation.