Dual phase convolutional neural network based system aimed at small rice grain dataset for disease identification

Tashin Ahmed [1], Chowdhury Rafeed Rahman[1] Md. Faysal Mahmud Abid[1]

[1]United International University
Dhaka, Bangladesh
tahmed153109@bscse.uiu.ac.bd, rafeed@cse.uiu.ac.bd, mabid143108@bscse.uiu.ac.bd

Abstract

Although Convolutional Neural Networks (CNNs) are widely used for plant disease detection, they require a large number of training samples while dealing with wide variety of heterogeneous background. In this paper, a CNN based dual phase method has been proposed which can work effectively on small rice grain disease dataset with heterogeneity. At the first phase, Faster RCNN method is applied for cropping out the significant portion (rice grain) from an image. This initial phase results in a secondary dataset of rice grains devoid of heterogeneous background. Disease classification is performed on such derived and simplified samples using CNN architectures. Comparison of the dual phase approach with straight forward application of CNN on the small grain dataset shows the effectiveness of the proposed method which provides a 5 fold cross validation accuracy of 88.11%.

Introduction

As rice grain diseases occur at the very last moment ahead of harvesting, it does major damage to the cultivation process. The average loss of rice due to grain discolouration (Baite et al. 2019) was 18.9% in India. In Bangladesh False Smut was one of the most destructive rice grain disease (Nessa 2017) from year 2000 to 2017. Collecting field level data on agronomy is a challenging task in the context of poor and developing countries. The challenges include lack of equipment and specialists. Farmers of such areas are ignorant of technology use which makes it quite difficult to collect crop disease related data efficiently using smart devices via the farmers. Hence, scarcity of plant disease oriented data is a common challenge while automating disease detection in such areas.

Many researches have been undertaken with a view to automating plant disease detection from the very beginning of deep learning revolution utilizing different techniques of machine learning and image processing. They applied a threshold based clustering algorithm for this task. A framework have been created (Sethy, Negi, and Bhoi 2017) for the detection of defected diseased leaf using K-Means clustering based segmentation. They claimed that their approach was able to detect the healthy leaf area and defected diseased area accurately. A genetic algorithm has been developed (Chung et al. 2016) which was used for selecting essential traits and optimal model parameters for the SVM classifiers for Bakanae *gibberella fujikuroi* disease.

Copyright © 2023, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

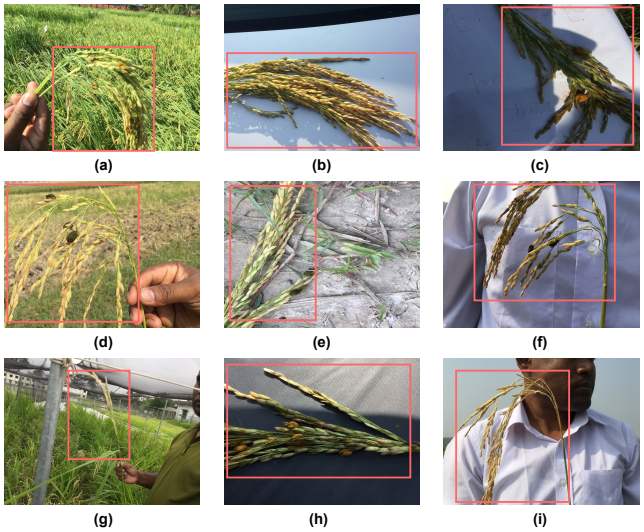

Figure 1: Sample of data heterogeneity amongst the dataset which demonstrates different parameters like unique backgrounds, lights, contrast, distance have been taken into account while data was gathered. Red rectangular boxes show how the annotation process has been performed. (a, d, f): multiclass images consists of Neck Blast and False Smut, (b): healthy, (c, e, h): False Smut, (g, i): Neck Blast.

A technique to classify the diseases based on percentage of RGB value of the affected portion was proposed (Islam et al. 2018) utilizing image processing. A similar technique using multi-level colour image thresholding was proposed (Bakar et al. 2018) for RLB disease detection. Deep learning based object classification and segmentation has become the state-of-the-art for automatic plant disease detection. Researchers also experimented with AlexNet (Atole and Park 2018) to distinguish among 3 classes of rice disease using a small dataset containing 227 images. A similar research for classifying 10 classes of rice disease on a 500 image dataset was undertaken (Lu et al. 2017) using a handmade deep CNN architecture. Furthermore, the benefit of us-

ing pre-trained model of AlexNet and GoogleNet (Brahimi, Boukhalfa, and Moussaoui 2017) has been demonstrated when the training data is not large. Their dataset consisted of 9 diseases of tomatoes. A detailed comparative analysis of different state-of-the-art CNN baseline and finely tuned architecture (Rahman et al. 2018) on eight classes of rice disease and pest also conveys a huge potential. It demonstrates two-stage training approach for memory efficient small CNN architectures. Besides works on rice disease some experimental procedure on other agricultural elements are, (Ferentinos 2018) specialized deep learning models based on specific CNN architectures for identification of plant leaf diseases with a dataset containing 58 classes from 25 different plants have been developed. Transfer learning approach using GoogleNet (Barbedo 2018) on a dataset containing 87848 images of 56 diseases infecting 12 plants. Faster RCNN has been applied (Bari et al. 2021) in a real-time approach to diagnose rice leaf disease, despite the dataset not being natural scene images.

Though the above mentioned researches have a significant contribution to the automation of disease detection, none of the works addressed the problem of scarcity of (natural scene) data which limits the performance of CNN based architectures for creating a productive solution. Most of the researches focused on image augmentation techniques to tackle the dataset size issue. But applying different geometric augmentations on small size images (Liu and Gillies 2016; Shorten and Khoshgoftaar 2019) result in nearly the same type of image production which has drawbacks in terms of neural network training. Production of similar images through augmentation (Cogswell et al. 2015) can cause overfitting as well.

The first phase of the proposed method deals with a learning oriented segmentation based architecture. This architecture helps in detecting the significant grain portion of a given image that has a heterogeneous background, which is an easier task compared to disease localization. The detected grain portions cropped from the original image are used as separate simplified images. In the second phase, these simplistic grain images are used in order to detect grain disease using fine tuned CNN architecture. Because of the simplicity of the tasks assigned in the two phases, our proposed method performs well in spite of having only 200 images of three classes. To prove the proposed approach is satisfactory for a small dataset, counter experimentations on straightforward CNN and Faster RCNN were also demonstrated in **Appendix D**.

Our Dataset

Our balanced dataset of 200 images consists of three classes - False Smut, Neck Blast and healthy grain class as shown in Table 1.

A sample image from each class has been shown in Fig. 1. Neck Blast is generally caused by the fungus known as *Magnaporthe oryzae*. It causes plants to develop very few or no grains at all. Infected nodes result (Wilson and Talbot 2009) in panicle break down. False Smut is caused by the fungus called *Ustilaginoidea virens*. It results in lower grain weight and reduction (Koiso et al. 1994) of seed germination.

Class	Image Count	Image Percentage
False Smut	75	37.5
Neck Blast	63	31.5
Healthy	62	31.0

Table 1: Image count and percentage(%) of each class of the primary dataset.

For this experiment, data has been obtained and annotated from two different sources: image data from a repository (Rahman et al. 2018) that has undergone previous testing and field data collected under the supervision of staff from the Bangladesh Rice Research Institute (BRRI). The Boro rice plant has been chosen for experimental data collection because Boro species pose the greatest risk (Miah et al. 1985) of being negatively impacted by Neck Blast and False Smut. When taking pictures, factors like light, distance, and uniqueness were taken into account. Heterogeneity of the background was the primary factor that was considered. Some sample images of hetergenous background of the dataset have been presented in Fig. 1. **Appendix A** provides additional explanations of the dataset.

Proposed Dual Phase Approach

In this study, a dual phase approach has been used to learn efficiently from a small dataset of images with significant background heterogeneity. Fig. 2 shows an overview of the approach. The initial stage involves taking the original image, cropping it to a specific size, and then running it through a segmentation-focused Faster RCNN architecture. From the first phase, two best regions have been chosen at most. The regions are cropped and resized to a fixed size after the significant grain portions of an image have been extracted. Because the backgrounds in these images are homogeneous, they appear straightforward. This condensed dataset has been used to train the CNN architecture for disease detection. Through experiments, the effectiveness of the learning process has been demonstrated.

Segmenting Grain Portion

This is the first stage of our approach. Images must be a fixed size to be used with segmentation algorithms that use CNN architecture as the backbone. Before feeding input images to the Faster RCNN, they have been scaled down to 640×480. Network's sequential stages through which this resized image travels are described in **Appendix B**.

Disease Detection from Segmented Grain

The Faster RCNN architecture is shown drawing bounding boxes on two significant grain portions in Fig. 2. Each of these frames is cropped and resized to a fixed size (in this case, 300×250) before being passed through a CNN architecture. As a result, the single image from the primary dataset has been divided into no more than two images. Each image in the primary dataset go through the same procedure. This allows for the creation of a secondary dataset

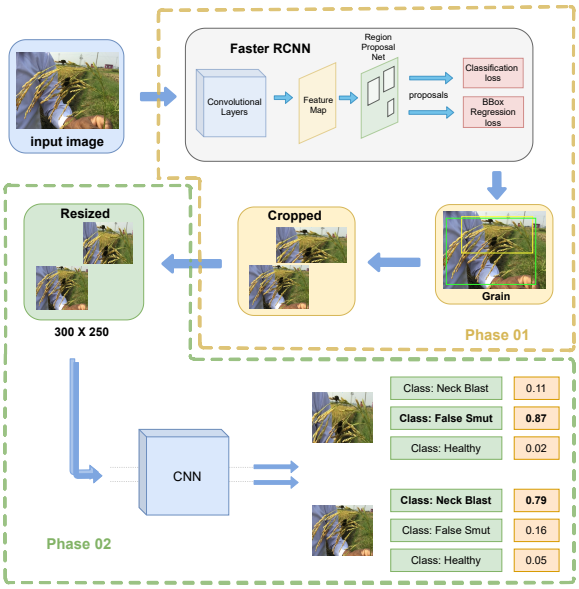

Figure 2: Proposed dual phase approach; Phase one for detection of the significant portion and phase two for classification; A multiclass data have been presented as an example to demonstrate the classification strategy.

Class	Image Count		Image
	Primary	Secondary	Increment
False Smut	75	85	10
Neck Blast	63	70	7
Healthy	62	64	2
Total	200	219	19

Table 2: Complete Dataset

with significant grain portion. To train the CNN architecture, each of these images must be assigned to one of the three classes. The full dataset, which includes these secondary image counts, is displayed in Table 2. As a multiclass data example, the cropped regions in Fig. 2 were predicted by a trained CNN model to represent False Smut and Neck Blast class.

Experimental Setup

Different hyperparameter optimization techniques have been covered in this section along with their effects on the experiment. The hardware used in the experiment, a description of the five basenets used, and evaluation metrics are all described in the **Appendix C** along with the experimental setup.

Optimized Hyperparameters

Hyperparameters of Faster RCNN have been presented in Table 5 in **Appendix C**.

Anchor Box Hyperparameters: Anchor boxes are a set of bounding boxes defined through different scales and aspect ratios. They mark the probable regions of interest of different shapes and sizes. The total number of probable anchor boxes per pixel of a convolutional feature map is $P_n \times R_n$, where P_n and R_n denote the number of anchor box size variations and ratio variations respectively.

Region Proposal Network (RPN) Hyperparameters: RPN layer utilizes the convolutional feature map of the original image to propose regions of interest that are identifiable within the original image. The proposals are made in line with the anchor boxes. For each anchor box, RPN predicts if it is an object of interest or not and changes the size of the anchor box to better fit the object. RPN threshold of 0.4 - 0.8 means that any proposed region which has IoU (Intersection Over Union) less than 0.4 with ground truth object is considered a wrong guess whereas any proposed region which has IoU greater than 0.8 with ground truth object is considered correct. This notion is used for training the RPN layer.

Proposal Selection: Proposal selection threshold of 200 means that top (according to probability) 200 region proposals from RPN layer will pass on to the next layers for further processing.

Overlap Threshold: During non-max suppression, overlapping object proposals are excluded if the IoU is above a certain threshold. If their overlap is greater than the threshold, only the proposal with the highest probability is kept and the procedure continues until there are no more boxes with sufficient overlap.

Learning Rate: It is used for controlling the speed of model parameter update.

Optimizer: Optimizer is an algorithm for updating model parameter weights after training on each batch of samples. Weight updating process varies with the choice of optimizer.

As the proposed method has two stages: segmentation and classification, failure in proper segmentation can leads to classification failure. In this work, two stages are created as an intact pipeline so that outcome of the first stage directed to the second stage as input. Detail about the procedure mentioned in Subsection Prime Experiment.

Results and Discussion

The proposed dual-phase approach has been carried out in two steps, as it was previously mentioned. First, the grain parts are segmented, and then the segmented parts are classified. Throughout the paper, this experiment has been referred to as the prime experiment. Different CNN architectures and the Faster RCNN have each been used separately to test the performance of the prime experiment. This part of the experiment has been mentioned as the counter experiment described in **Appendix D**.

Prime Experiment: Dual Phase Approach

Prime experiment has been performed by creating a pipeline in two phases shown in Fig. 2. In the first phase, segmentations of grain has been performed and the segmented parts have been cropped and saved as the secondary dataset, mentioned in Subsection Disease Detection from Segmented

Anchor Box Ratio	Anchor Box Pixels	RPN Threshold	CNN Architecture	Overlap Threshold	mAP (%)
(1:1),($\frac{1}{\sqrt{2}}$: $\frac{2}{\sqrt{2}}$), ($\frac{2}{\sqrt{2}}$: $\frac{1}{\sqrt{2}}$),(2:2))	32, 64, 128, 256	0.4 - 0.8	**VGG16**	**>0.8**	**84.3 ± 2.36**
			VGG19	>0.8	73.5 ± 1.34
			ResNet50	>0.8	65.9 ± 2.59

Table 3: Prime Experiment: Phase One

CNN Architecture	Train Loss	Train Accuracy (%)	Validation Loss	Validation Accuracy (%)
VGG16	**0.196**	**94.47**	**0.195**	**88.11 ± 3.86**
VGG19	0.095	89.98	0.093	86.43 ± 2.98
ResNet50	0.367	89.63	0.281	78.00 ± 2.32

Table 4: Prime Experiment: Phase Two

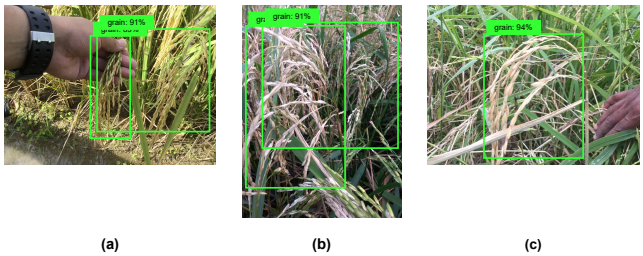

Figure 3: Prime Experiment: First Phase; Sample Outcome: accuracy of detected objects. Phase One detected two bounding boxes from (a) and (b) as the second bounding box meets IoU and accuracy threshold. (c) have not met the IoU or accuracy threshold so the second bounding box is absent.

Grain. K fold cross-validation (K=5) have been performed where train and validation split was 80:20. As a result, the full primary dataset has been converted into a secondary dataset. After labeling the secondary dataset into three classes, three chosen CNN architectures have been used in second phase to perform the final classification.

Phase One: Segmentation of Grains To remove the significant portion (grain) from a specific image is the goal of phase one. Three different CNN architectures have been used as the backbone, with Faster RCNN being used in between. VGG16, VGG19, and ResNet50 CNN architectures were used, which have also been used in counter experiment 02, mentioned in Subsection Counter Experiment 02. Also, hyperparameters setting have been followed from the counter experiment 02. Only the best performed hyperparameter setting were applied in phase one mentioned in Table 5. Faster RCNN with VGG16 as backbone achieved the best mAP score of 84.3 ± 2.36. The outcome has been obtained using 5-fold cross validation. As a result, every image in the dataset has been assessed. At most two new images could be generated from each original image, resulting in the creation of a secondary dataset. Two of each image's best bounding boxes have been chosen to perform this operation. The first bounding box, which is cropped and included in the new dataset, is the best bounding box outlined by the Faster RCNN. Second bounding boxes that meet the accuracy threshold of 90% and IoU threshold of 0.5 have been chosen. Multiple images lacked bounding boxes that concurred with this requirement. In that instance, only one image has been chosen for the new dataset. The bounding boxes from each phase one image are shown in Fig. 3. Only one bounding box is detected in this instance on subfigure (c).

Phase Two: Classification Phase two is where the classification result is produced using the image data from phase one. In this phase, three different CNN architectures VGG16, VGG19, and ResNet50 have once more been used. In the phase referred to **Appendix D**, the best settings from counter experiment 01 have been reapplied. With a validation accuracy of 88.11 ± 3.86, VGG16 stood out as having the best performance mentioned in Table 4.

Conclusion and Future Work

In a nutshell, the following contributions are made by this research: A dual phase strategy that can learn from a small dataset on rice grain diseases has been put forth. Phase one has proposed a smart segmentation method that can handle the heterogeneous background present in the dataset of plant disease images, that was gathered in a real-world setting. On a small dataset of rice grain images, an experimental comparison with the use of current CNN architectures has been offered to demonstrate the efficacy of the suggested method.

Since there are only 3 classes and they differ greatly in terms of class features, expectancy from CNN models like VGG16 is higher in this dataset. This process is pipeline-based, and phase one may produce FP/FN results that are passed on to phase two. Phase 2 will not be able to properly classify them as a result. That is one drawback of this system. This problem can be resolved by introducing a new class called "No Grain" that will declare everything except grain.

References

Atole, R. R.; and Park, D. 2018. A Multiclass Deep Convolutional Neural Network Classifier for Detection of Common Rice Plant Anomalies. *INTERNATIONAL JOURNAL OF ADVANCED COMPUTER SCIENCE AND APPLICATIONS*, 9(1): 67–70.

Baite, M. S.; Raghu, S.; Prabhukarthikeyan, S.; Keerthana, U.; Jambhulkar, N. N.; and Rath, P. C. 2019. Disease incidence and yield loss in rice due to grain discolouration. *Journal of Plant Diseases and Protection*, 1–5.

Bakar, M. A.; Abdullah, A.; Rahim, N. A.; Yazid, H.; Misman, S.; and Masnan, M. 2018. Rice leaf blast disease detection using multi-level colour image thresholding. *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, 10(1-15): 1–6.

Barbedo, J. G. A. 2018. Impact of dataset size and variety on the effectiveness of deep learning and transfer learning for plant disease classification. *Computers and electronics in agriculture*, 153: 46–53.

Bari, B. S.; Islam, M. N.; Rashid, M.; Hasan, M. J.; Razman, M. A. M.; Musa, R. M.; Ab Nasir, A. F.; and Majeed, A. P. A. 2021. A real-time approach of diagnosing rice leaf disease using deep learning-based faster R-CNN framework. *PeerJ Computer Science*, 7: e432.

Brahimi, M.; Boukhalfa, K.; and Moussaoui, A. 2017. Deep learning for tomato diseases: classification and symptoms visualization. *Applied Artificial Intelligence*, 31(4): 299–315.

Chollet, F. 2017. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1251–1258.

Chung, C.-L.; Huang, K.-J.; Chen, S.-Y.; Lai, M.-H.; Chen, Y.-C.; and Kuo, Y.-F. 2016. Detecting Bakanae disease in rice seedlings by machine vision. *Computers and electronics in agriculture*, 121: 404–411.

Cogswell, M.; Ahmed, F.; Girshick, R.; Zitnick, L.; and Batra, D. 2015. Reducing overfitting in deep networks by decorrelating representations. *arXiv preprint arXiv:1511.06068*.

Ferentinos, K. P. 2018. Deep learning models for plant disease detection and diagnosis. *Computers and Electronics in Agriculture*, 145: 311–318.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Islam, T.; Sah, M.; Baral, S.; and RoyChoudhury, R. 2018. A Faster Technique on Rice Disease Detectionusing Image Processing of Affected Area in Agro-Field. In *2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT)*, 62–66. IEEE.

Koiso, Y.; Li, Y.; Iwasaki, S.; HANAKA, K.; Kobayashi, T.; Sonoda, R.; Fujita, Y.; Yaegashi, H.; and Sato, Z. 1994. Ustiloxins, antimitotic cyclic peptides from false smut balls on rice panicles caused by Ustilaginoidea virens. *The Journal of antibiotics*, 47(7): 765–773.

Liu, R.; and Gillies, D. F. 2016. Overfitting in linear feature extraction for classification of high-dimensional image data. *Pattern Recognition*, 53: 73–86.

Lu, Y.; Yi, S.; Zeng, N.; Liu, Y.; and Zhang, Y. 2017. Identification of rice diseases using deep convolutional neural networks. *Neurocomputing*, 267: 378–384.

Miah, S.; Shahjahan, A.; Hossain, M.; and Sharma, N. 1985. A survey of rice diseases in Bangladesh. *International Journal of Pest Management*, 31(3): 208–213.

Nessa, B. 2017. *Rice False Smut Disease in Bangladesh: Epidemiology, Yield Loss and Management*. Ph.D. thesis, PhD thesis, Department of Plant Pathology and Seed Science, Sylhet ....

Rahman, C. R.; Arko, P. S.; Ali, M. E.; Khan, M. A. I.; Apon, S. H.; Nowrin, F.; and Wasif, A. 2018. Identification and recognition of rice diseases and pests using convolutional neural networks. *arXiv preprint arXiv:1812.01043*.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, 91–99.

Sethy, P. K.; Negi, B.; and Bhoi, N. 2017. Detection of healthy and defected diseased leaf of rice crop using K-means clustering technique. *International Journal of Computer Applications*, 157(1): 24–27.

Shorten, C.; and Khoshgoftaar, T. M. 2019. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1): 60.

Simonyan, K.; and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; and Wojna, Z. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2818–2826.

Wilson, R. A.; and Talbot, N. J. 2009. Under pressure: investigating the biology of plant infection by Magnaporthe oryzae. *Nature Reviews Microbiology*, 7(3): 185–195.

Appendix A. Additional Informations on Dataset

To make the dataset more challenging multiclass images also have been taken into account. Sample images of multiclass data which consists both Neck Blast and False Smut class have been presented in Fig. 4. If there is a multiclass data it had been labelled as Neck Blast for the training phase of counter experiments (explained in Section Results and Discussion) as False Smut already surpassed by quantity than Neck Blast. Also, the dataset was split into 80:20 in terms of train and validation set. Augmentation techniques were not applied as the main goal of this experiment is to use small and natural scene image data. Additionally, there are other factors that can spoil the experiment, such as illumination, symptom severity, maturity of the plant and diseases. A large versatile dataset can attend on that occasion which can be achieved in the future. The dataset has been kept to three classes for the early stages of the investigation. Also, it is quite challenging and burdensome to collect different rice disease image dataset throughout the year as different diseases occur at different time. So, at this early stage of the investigation three classes is competent to deliver a sufficient result. Supplementary public data related to the paper can be found at https://cutt.ly/RvgxoDi.

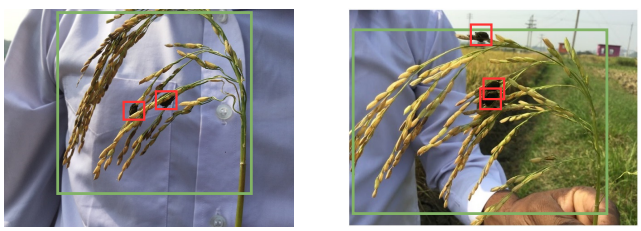

Figure 4: Sample multiclass data; Labeled green for Neck Blast and red for False Smut class

Appendix B. Consecutive Stages of Phase One

Convolutional Neural Network (CNN): In order to avoid sliding a window in each spatial position of the original image, CNN architecture is used in order to learn and extract feature map from the image which represents the image effectively. The spatial dimension of such feature map decreases whereas the channel number increases. For the dataset used in this research, VGG16 architecture has proven to be the most effective. Hence, VGG16 has been used as the backbone CNN architecture which transforms the original image into $20 \times 15 \times 512$ dimension.

Region Proposal Network (RPN): The extracted feature map is passed through RPN layer. For each pixel of the feature map of spatial size 20×15, there are 16 possible bounding boxes (4 different aspect ratios and 4 different sizes mentioned in bold letter in Table 5). So, that makes total 16×20×15 = 4800 possible bounding boxes, RPN is a two branch Convolution layer which provides two scores (branch one) and four coordinate adjustments (branch two) for each of the 4800 boxes. The two scores correspond to the probability of being an object and a non-object. Only those boxes which have a high object probability are taken into account. Non-max suppression (NMS) is used in order to eliminate overlapping object bounding boxes and to keep only the high probability unique boxes. The threshold of this overlap in this case is 0.8 IoU. From this probable object proposals, top 200 proposals according to object probability are passed to the next layers.

ROI Pooling: Each of the 200 selected object proposals correspond to some region in the CNN feature map. For passing each of these regions on to the dense layers of the architecture, each of the regions need to be of fixed size. ROI pooling layer takes each region and turns them into 7×7×512 using bilinear interpolation and max pooling.

RCNN Layer: RCNN layer consists of fully connected dense layers. Each of the 7×7×512 size feature maps are flattened and passed through these fully connected layers. The final layer has two branches. Branch one predicts if the input feature map is background class or significant grain portion. Branch two provides four regression values denoting the adjustment of the bounding box to better fit the grain portion. For each feature map, if the probability of being a grain is over 0.6, only then is the feature map considered as a probable grain portion and the adjusted coordinates are mapped to the original image in order to get the localized grain portion. The overlapping boxes are eliminated using NMS. The remaining bounding box regions are the significant grain portions.

Loss Function: The trainable layers of Faster RCNN architecture are: CNN backbone, RPN layer and RCNN layer. A loss function is needed in order to train these layers in an end to manner which is as follows.

$$L(p_i, t_i) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*) \tag{1}$$

The first term of this loss function defines the classification loss over two classes which describe whether predicted bounding box i is an object or not. The second term defines the regression loss of the bounding box when there is a ground truth object having significant overlap with the box. Here, p_i and t_i denote predicted object probability of bounding box i and predicted four coordinates of that box respectively while p_i^* and t_i^* denote the same for the ground truth bounding box which has enough overlap with predicted bounding box i. N_{cls} is the batch size (256 in this case) and N_{reg} is the total number of bounding boxes having enough overlap with ground truth object. Both these terms work as normalization factor. L_{cls} and L_{reg} are log loss (for classification) and regularized loss (for regression) function respectively.

Appendix C. Hardware, Utilized CNN Models and Evaluation Metrics

For the training environment, assistance has been taken from two different sources.

- Royal Melbourne Institute of Technology (RMIT) provides GPU for international research enthusiasts and they provided a Red Hat Enterprise Linux Server along with the processor Intel Xeon E5-2690 CPU, clock speed of 2.60 GHz. It has 56 CPUs with two threads per core, 503 GB of RAM. Each user can use up to 1 petabyte of storage. There are also two 16 GB NVIDIA Tesla P100-PCIE GPUs available. First phase was completed through this server.
- Google Colab (Tesla K80 GPU, 12GB RAM) and Kaggle kernel (Tesla P100 GPU) have been used for counter experimentation.

Fig. 5 shows architectures and key blocks of the applied CNN architectures. Experiments have been performed using five state-of-the-art CNN architectures which are described as follows.

VGG16 is a sequential architecture (Simonyan and Zisserman 2014) consisting of 16 convolutional layers. Kernel size in all convolution layers is three.

VGG19 has three extra convolutional layers (Simonyan and Zisserman 2014) and the rest is the same as VGG16.

ResNet50 belongs to the family of residual neural networks. It is a deep CNN architecture (He et al. 2016) with skip connections and batch normalization. The skip connections help in eliminating the gradient vanishing problem.

InceptionV3 is a CNN architecture (Szegedy et al. 2016) with parallel convolution branching. Some of the branches have filter size as large as 7×7.

Xception takes the principles of Inception to an extreme. Instead of partitioning the input data into several chunks, it maps the spatial correlations (Chollet 2017) for each output channel separately and performs 1×1 depthwise convolution.

All results have been provided in terms of 5 fold cross validation. Accuracy metric has been utilized in order to compare dual phase approach against implementation of CNN on original images without any segmentation. Accuracy is a suitable metric for balanced dataset.

$$Accuracy = \frac{TP}{TP + FP + TN + FN}^{1} \tag{2}$$

Segmenting the grain portion is the goal of the first phase of the dual phase approach. For evaluating the performance of this phase, mAP (mean average precision) score has been used. Precision, recall and IoU (Intersection over Union) are required to calculate mAP score.

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

[1]TP: True Positive, FP: False Positive, TN: True Negative, FN: False Negative

$$Recall = \frac{TP}{TP + FN} \tag{4}$$

$$IoU = \frac{AOI}{AOU}^{2} \tag{5}$$

If a predicted box IoU is greater than a certain predefined threshold, it is considered as TP. Otherwise, it is considered as FP. $(TP + FN)$ is actually the total number of ground truth bounding boxes. Average precision (AP) is calculated from the area under the precision-recall curve. If there are N classes, then mAP is the average AP of all these classes. In this research, there is only one class of object in phase one, that is the significant grain portion class. So, here AP and mAP are the same.

Appendix D. Counter Experiments

Individual counter experiments have been performed to analyze their performance with the respective phase of the prime experiment.

Counter Experiments

There have been two counter experiments, counter experiment 01 and counter experiment 02. The objective of counter experiment 01, which is based on different CNN architectures, is to obtain the classification result from the main dataset. The second counter experiment, which has been used for both classification and detection of the three classes, is based on a Faster RCNN that underpins three chosen CNN architectures.

Counter Experiment 01: CNN In this experiment, five different CNN architectures were used. In this section, using imagenet pretrained models, three transfer learning methodologies, freezed layer, fine tuning, and fine tuning + dropout have been applied. The freezing layer approach, also known as the default transfer learning method, was used initially. With a validation accuracy of 63.33 ± 2.04, VGG16 performed better than other CNN architectures. After that, a finely tuned approach has been used, which resulted in an increase in validation accuracy for VGG16 of 67.79 ± 3.24. The CNN architectures have been modified to incorporate dropout, which yields a significant improvement of 69.43 ± 3.41 for VGG16. By experimenting with dropout on different positions inside individual CNNs, fine-tuning and fine-tuning + dropout have both been performed repeatedly. Although the VGG16 results in high standard deviation, which is a sign of poor precision. In Table 6, comparative results for each of the five architectures are displayed.

Counter Experiment 02: Faster RCNN Faster RCNN has been employed in this counter experiment 02, with three different CNN architectures serving as the backbone. The purpose of this experiment is to evaluate Faster RCNN's capability for effective significant portion detection and classification (grain). The effectiveness of VGG16 and VGG19

[2]AOI: Area of intersection, AOU: Area of union (with respect to ground truth bounding box)

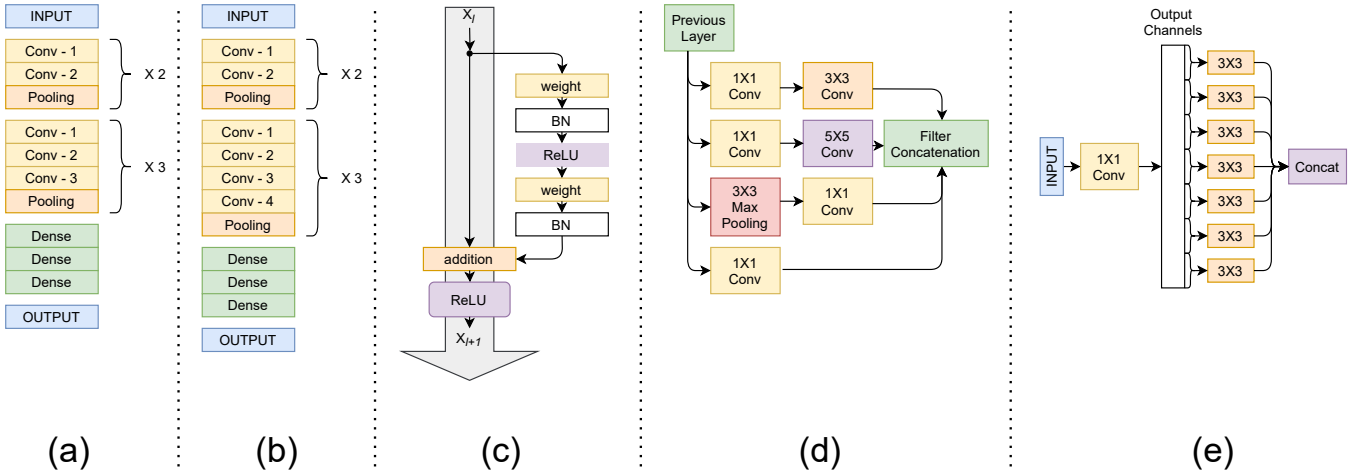

Figure 5: (a): all 16 blocks of VGG16, (b): all 19 blocks of VGG19, (c): Resdiual module of ResNet, (d): Inception module is to act as a multi-level feature extractor in InceptionV3, (e): Extreme module of the Inception module which is utlized in Xception.

Hyperparameter	**Optimized Value**
Anchor Box Count	9, **16**
Anchor Box Size (pixels)	[**32, 64, 128, 256**], [128, 256, 512]
Anchor Box Ratios	[(1,1), (2,1), (1,2)], [**(1,1), ($\frac{1}{\sqrt{2}}$, $\frac{2}{\sqrt{2}}$), ($\frac{2}{\sqrt{2}}$, $\frac{1}{\sqrt{2}}$), (2,2)**]
RPN Threshold	0.3 - 0.7, **0.4 - 0.8**
Proposal Selection	**200**, 2000
Overlap Threshold	>**0.8**, >0.9
Learning Rate	0.001, **0.0001**, 0.00001
Optimizers	**Adam**, SGD

Table 5: Experimented hyperparameters. Bold values were selcted for the prime experiment.

Transfer learning Approach	CNN Architecture	Validation Loss	Validation Accuracy (%)
Freezed Layer	**VGG16**	**2.08**	**63.33 ± 2.04**
	VGG19	1.08	43.75 ± 3.43
	Xception	2.34	31.25 ± 4.04
	InceptionV3	9.23	37.50 ± 3.89
	ResNet50	4.47	31.25 ± 3.27
Fine Tuned	**VGG16**	**2.71**	**67.79 ± 3.24**
	VGG19	1.77	55.04 ± 3.00
	Xception	6.29	43.76 ± 1.88
	InceptionV3	7.42	41.73 ± 3.66
	ResNet50	2.47	38.20 ± 1.34
Fine Tuned + Dropout	**VGG16**	**3.47**	**69.43 ± 3.41**
	VGG19	3.11	57.18 ± 2.64
	Xception	5.72	47.17 ± 2.11
	InceptionV3	4.12	48.22 ± 3.14
	ResNet50	2.81	42.31 ± 1.32

Table 6: Counter Experiment 01: CNN

has been determined by counter experiment 01. Additionally, ResNet50 was chosen over Xception and InceptionV3 (which are mentioned in Table 6) due to the lower validation loss. CNN models that were pretrained using data from COCO and Imagenet have been used. For the Faster RCNN model, various hyperparameter optimizations have been used to achieve the best results, which are shown in Table 7.

Default settings from the Faster RCNN paper (Ren et al. 2015) for anchor box ratio were (1:1), (2:1), (1:2) and anchor box pixels were 128, 256, 512. Which produces 3×3=9 anchor boxes per pixel. The default RPN threshold of (0.3 - 0.7), overlap threshold 0.8 and default anchor box ratios and pixels, VGG16 (imagenet pretrained model) provided the best mAP score of 71.0±4.0. After tuning RPN threshold to (0.4 - 0.8), anchor box ratios to (1:1), ($\frac{1}{\sqrt{2}}$: $\frac{2}{\sqrt{2}}$), ($\frac{2}{\sqrt{2}}$: $\frac{1}{\sqrt{2}}$), (2:2) and pixel sizes to 32, 64, 128, 256, 4×4=16 anchor boxes have been produced which provides better outcome than before. This setting improved the mAP for VGG16 (imagent pretrained model) to 76.32 ± 2.29 which is the peak outcome after several optimization.

Pretrained Model	Anchor Box Ratio	Anchor Box Pixels	RPN Threshold	CNN Architecture	Overlap Threshold	mAP (%)
Imagenet	(1:1), (2:1), (1:2)	128,256, 512	0.3 - 0.7	VGG16	>0.8	71.0 ± 4.0
				VGG19		47.06 ± 2.01
				ResNet50		67.14 ± 6.68
	(1:1), ($\frac{1}{\sqrt{2}}$: $\frac{2}{\sqrt{2}}$), ($\frac{2}{\sqrt{2}}$: $\frac{1}{\sqrt{2}}$), (2:2)	32, 64, 128, 256	0.4 - 0.8	**VGG16**	**>0.8**	$\mathbf{76.32 \pm 2.29}$
					>0.9	63.42 ± 2.36
				VGG19	>0.8	70.08 ± 4.54
					>0.9	70.30 ± 2.36
				ResNet50	>0.9	40.02 ± 3.03
					>0.8	52.36 ± 5.91
COCO	(1:1), (2:1), (1:2)	128,256, 512	0.3 - 0.7	VGG16	>0.8	48.32 ± 4.79
				VGG19		32.30 ± 4.83
				ResNet50		46.36 ± 2.04
	(1:1), ($\frac{1}{\sqrt{2}}$: $\frac{2}{\sqrt{2}}$), ($\frac{2}{\sqrt{2}}$: $\frac{1}{\sqrt{2}}$), (2:2)	32, 64, 128, 256	0.4 - 0.8	**VGG16**	**>0.8**	$\mathbf{54.24 \pm 2.23}$
					>0.9	48.0 ± 2.10
				VGG19	>0.8	42.36 ± 1.02
					>0.9	41.07 ± 5.21
				ResNet50	>0.9	28.42 ± 4.84
					>0.8	30.23 ± 3.0

Table 7: Counter Experiment 02: Faster RCNN