International Journal on Advanced Science Engineering Information Technology

Vol.12 (2022) No. 3
ISSN: 2088-5334

Classification of Air-Cured Tobacco Leaf Pests Using Pruning Convolutional Neural Networks and Transfer Learning

Dwiretno Istiyadi Swasono [a,b,*], Handayani Tjandrasa [a], Chastine Fatichah [a]

[a] *Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, 60111, Indonesia*
[b] *Department of Computer Science University of Jember, Jember, 68124, Indonesia*
Corresponding author: [*]istiyadi@unej.ac.id

***Abstract*— Convolutional Neural Network (CNN) usually uses a large image dataset with many parameters. Small datasets require a small number of parameters. Existing standard (pre-trained) models such as Alexnet, VGG, Inception, and Resnet have been tested with high accuracy but have many parameters. For small datasets, too many parameters become less efficient and increase computation costs. The high computational costs make the model unsuitable for computers with limited resources such as embedded devices and mobile phones. This research proposes pruning on the depth of resnet50 architecture and adds a dimensionality reduction layer after the pruning point. This approach does not require a complex pruning criteria algorithm, so it is easy to implement. Resnet50 was chosen because it is a good performance with batch normalization and skip connections. We use transfer learning for Resnet50 weight. Pruning is carried out at a depth of the network by cutting at the layer of the activation function. Several pruning points were selected to produce several models with certain parameters. The more networks layer pruned, the smaller the number of parameters produced. We add a layer for channel reduction after pruned network to reduce the number of feature maps before entering the fully connected (FC) layer as a classifier. We retrained a new network using a 2000 tobacco leaf pest dataset split into 1600 training and 400 validation images with 4-classes. The result shows that the accuracy could be maintained equal to the unpruned network up to 100% accuracy and 74.38% reduction rate for the number of parameters. A higher reduction rate of the number of parameters up to 90.62% still provides high accuracy of validation data around 99.3%. These prove that our proposed method effectively maintained accuracy and reduced the number of parameters.**

***Keywords*— Convolutional neural network; transfer learning; pruning; tobacco leaf pest.**

Manuscript received 2 Aug. 2021; revised 24 Sep. 2021; accepted 13 Dec. 2021. Date of publication 30 Jun. 2022.

I. INTRODUCTION

The agro-industry sector is still a mainstay that can support the economy and address the workforce in many countries. One of the agro-industry products is tobacco leaf. Tobacco is not only used in making cigarettes/cigars, but it can also be used as other raw materials such as perfume, biopesticides, and drugs. Its role as an industrial raw material demands high-quality tobacco leaves to produce products according to the expected standards. Processing tobacco leaves is a process that requires controlled environmental conditions and takes a relatively long time to several months. For example, processing tobacco leaves for cigar material.

Several factors can affect the quality of tobacco leaves. Pests and diseases can reduce the quality significantly. Leaves attacked by pests and diseases require to be separated from healthy leaves. Usually, the sorting process is done manually by human labor.

Manual sorting can lead to subjective results where different experts can rate the same object differently. Manual sorting is also prone to errors due to internal and external human factors. Internal factors, for example, are due to fatigue and external factors such as environmental lighting factors. Therefore, this study proposes a method of automatically sorting/classifying tobacco plant pests using computer vision.

Several previous studies have related pests and diseases and the quality of tobacco leaves. Marzan *et al.* [1] used a CNN classifier for grading tobacco leaves using segmented images. Harjoko *et al.* [2] used image preprocessing and color extraction for grading tobacco leaves using a threshold for the local color category and majority voting for the global color category. Guru *et al.* [3] detect the maturity level of tobacco leaves to be harvested using GLTP (Gray Level Local Texture Patterns). Sun *et al.* [4] identified tobacco leaf pests using multiple-attention modules and InceptionV3 transfer learning. Swasono *et al.* [5] classified tobacco leaf pests using VGG16

transfer learning with complete architecture or without pruning.

The object of this study is pest and diseases of air-cured tobacco leaves that have distinctive characteristics, i.e., glassy, green spot, frog skin, and thrip. For example, Glassy is marked by a rather wide area with blackish color and vague boundaries. Green Spot is a greenish area on the leaves that vary in size from small to rather large. The Frog Skin is black dots at certain adjacent distances and often becomes faint when photographed. Thrips are white dots clustered around the midrib that varies in numbers. In general, the base color of the leaves is tawny and usually larger than attacked area, but even small emergence of pests and diseases attacking the leaves is considered unacceptable. Several traditional methods for enhancement, segmentation, and feature extraction have tried to identify pests and diseases of tobacco plants. For example, color extraction using RGB (red green blue) and HSV (hue saturation value). For texture extraction using GLCM (Gray Level Co-occurrence Matrices) and LBP (Local Binary Pattern). Those methods gave an unsatisfactory level of accuracy to detect all types of pest and diseases attack. Different sizes, colors, textures between types of disease make extraction difficult enough to require integrated feature extraction and classification methods.

CNN has been chosen because it has many advantages over traditional image recognition methods. CNN, known as LeCun *et al*. [6], could train CNN to recognize handwritten characters. The architecture of CNN is growing to get better results. Some CNN architectures that are enough popular and widely developed are Googlenet/Inception [7] and Residual network [8]. We combine Resnet50 (for transfer learning and pruning) and use a part of the Inception module to build a CNN with high accuracy and reduce the number of parameters.

CNN training with high depth and an enormous number of parameters can take a long time to several days even though it uses Graphics Processor Unit (GPU) with thousands of parallel processor cores. It will be much longer if using a single core or a few CPU cores. In the course of the training, results can be transferred to recognize other objects that are different. Moving the learning result is known as transfer learning or fine-tuning. Using transfer learning can be done quickly because it uses the previous training results that are already convergent. The previous training result is a model used to initialize the initial network weight to be trained. It only needs a little extra layer that requires to be trained to match the number of the classes in the new datasets that will make convergence is very fast. Another advantage of transfer learning is an easy design because it does not need to redesign the overall CNN architecture but with little adaptation to the new dataset. The accuracy of the training results follows a well-used architecture.

This research proposes a classification of air-curing tobacco leaf pests using CNN with some modification in pruning and adding some layers. The used CNN architecture for transfer learning is the Residual network (Resnet), a state-of-the-art CNN architecture from 2015. Resnet has proven to outperform several architectures from Alexnet, VGGNet, to Inception on ILSVRC 2015 with the Imagenet dataset. Resnet has several variants. Resnet50 is a variant of Resnet, which has 50 convolutional layers. Other variants of Resnet that are commonly used are Resnet101 and Resnet152, which have convolutional layers and a higher total number of parameters. Resnet50 was chosen because it was considered sufficient to handle the tobacco leaf pest dataset, which was far smaller than the Imagenet dataset having 1.2 million image data and 1000 classes. With Resnet50 the resulting accuracy is very high and gives hope of CNN implementation on tobacco agroindustry in real terms.

However, the number of Resnet50 parameters is still quite high that above 23 million parameters. With such a large number of parameters, it is quite difficult to implement on mobile devices or computers with limited resources. Some research on CNN pruning has been done at feature maps or kernel level. For example, Zou *et al.* [9] pruned the low discriminability magnitude feature map on the VGGNet architecture. Li *et al.* [10] pruned convolutional filters considering the importance of interactive filters based on instability and sensitivity. Ayinde et al. pruned redundant features based on the relative cosine distance in the feature space [11]. Yang *et al*. [12] pruned redundant filters based on regularization and removed unimportant filters or made values to zero and some other studies [13]–[15], [23]. Generally, these studies perform pruning on redundant, sparsity, irrelevant, or unimportant filters/channels or feature maps in convolutional or fully connected layers using their knowledge method.

This study proposed pruning on the depth of Resnet50 transfer learning. The depth pruning of the Resnet50 network is possible to conduct because the Resnet50 network has a deep layer with approximately 50 for the convolutional layer. We add a dimensionality reduction layer between the pruning point and the FC layer. We use a 1x1 kernel convolutional layer in the Inception module [7]. It means that we reduce the number of features entering the FC layer. Depth pruning and reducing the number of features will significantly decrease the number of parameters. It makes the cost of computation much lower and more efficient. In addition, the network is smaller, so it is more suitable for computers with limited resources. We retrain the pruned network using the tobacco leaf pest dataset. This approach proves to keep the accuracy as high as the unpruned network method**.**

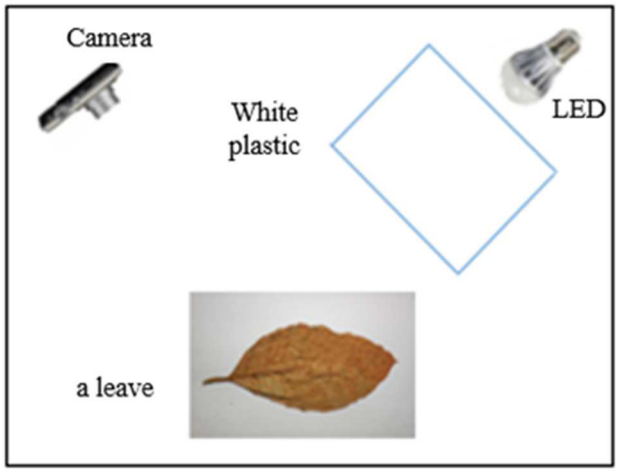
II. Material and Method.

A. Image Data Acquisition

Image data acquisition will determine the success of image recognition for the next. The image captured must be able to represent the information contained in it. The data acquisition of tobacco leaf images must be made carefully. The intensity of room light and the angle of taking can easily affect the image color leaf. Some features of the tobacco leaf pests can be lost/not visible if data acquisition fails. Often the image results are too dark even though the light used is bright enough. The image results were often not uniform in color and brightness if angles and lighting were not set correctly.

Therefore, this research tries to adjust the light intensity of the room and the angle of shooting to get the desired image. Lighting is done by using a 10Watt LED light and covered with a cloudy white plastic to scatter light. We expect to produce a more natural and gentler image using scattering light. Light scattering works like a softbox component in

photography techniques. The camera used is a Nikon series Coolpix S2500 with 4608x3456 pixels resolution. Fig. 1(a) shows the composition of the camera and equipment for data acquisition image and the image of the tobacco leaf on the curing stage, and Fig. 1(b) exemplifies some examples of tobacco leaf images.



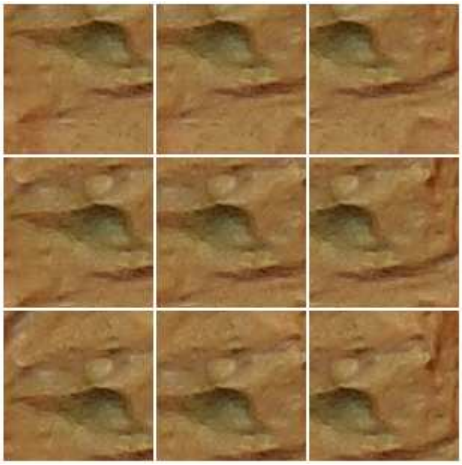
(a)

(b)

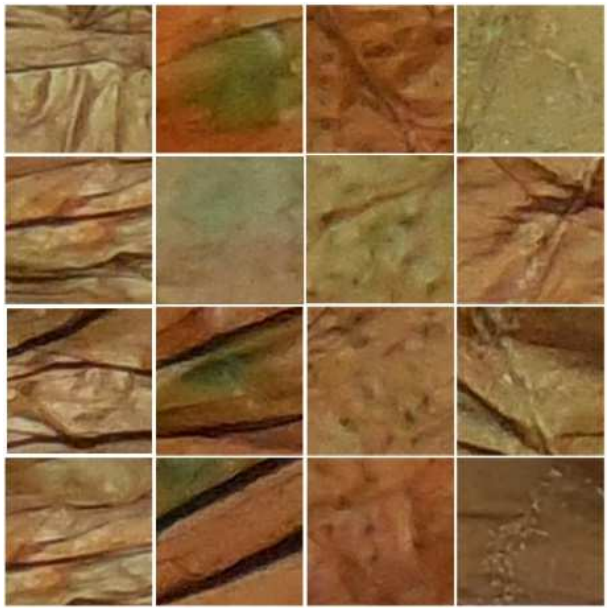
Fig. 1 Data acquisition: (a) the construction of camera, lights, and leaves (b) examples of air curing tobacco leaf images

Training and validation data do not use the whole leaves but use 224x224 pixel sub-images containing certain pest diseases with 2000 data. We divide the dataset into 1600 training data and 400 validation data. Sub-images on one leaf pest object are taken 9-times starting from the center of the object and shifting 20 pixels to the 8 positions at an angle of 0, 45, 90, up to 315 degrees. This way has the advantage because the number of images generated has more variations, such as the data augmentation process. Augmentation data is important for CNN to recognize objects that change due to translation, rotation, etc. The examples of sub-images with 20-pixels translations performed in 8-angle directions are shown in Fig. 2(a).

The size of 224x224 pixels is to adjust the image size used in Resnet-50. Fig. 2(b) shows examples of four types of pests of diseases used in this study, those are Glassy, Green Spot, Frog Skin, and Thrips. Glassy is characterized by more glossy leaf surface features than normal leaf surfaces. Green Spot is shown by the characteristics of a green stripe with a certain shape and size. Frog Skin is shown in the presence of black bits of darkness that often seem a little faint. Thrips are white dots around the leaf bone with a random pattern. These pests appear after the first fermentation stage and are unseen at the fresh leaf and after the curing stage.



(a)



(b)

Fig. 2 Training images: (a) data augmentation with 20-pixel translations (b) samples of tobacco leaf disease

B. Pruning Method

The proposed architectural block diagram of the training process is shown in Fig. 3 below. The Resnet50 Pretrained model has been trained with the Imagenet dataset. Resnet50 is suitable for transfer learning for a smaller dataset (such as the tobacco leaf pest dataset). The pruned network acts like feature extraction, and we add feature downsampling for feature reduction and FC layer as a classifier. The training process for this new network using training and validation of the tobacco leaf pest's dataset will produce a model that we can use for classification then.

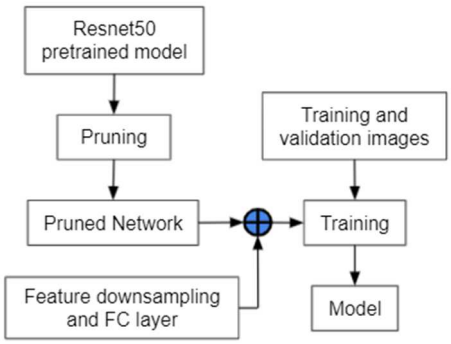


Fig. 3 Proposed training steps

The proposed pruning steps were performed by cutting the Resnet50 network on a particular layer. Several fundamental layers in Resnet50 can be selected as pruning points, namely convolution layers, batch normalization, and ReLU (Rectified Linear Unit) activation. The activation layer is selected because it is usually the final layer of a convolution layer unit, batch normalization, and activation (Fig. 4). In Resnet50 there are 49 activation layers with the ReLU activation function that are possibly chosen as pruning points. The ReLU activation function was signed by the number index that shows where it lied in the network, for example, activation_1, activation_2, up to activation_49. The number also shows the depth of the network. From the 49 pruning points, only some points were chosen as pruning points to accelerate training in new pruned networks. The training is tested at some point samples, for example is activation_9, activation_15, activation_24, activation_33, activation_42, and activation_49. Fig. 4 shows the number of parameters in each pruned network result for every pruning point index. The smaller index will prune more networks, and it will produce a lesser number of parameters.

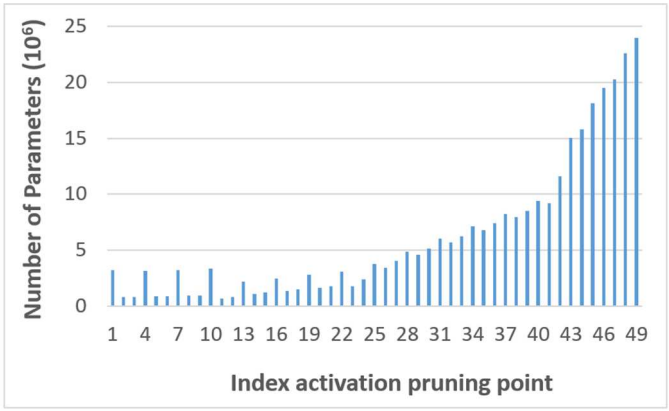
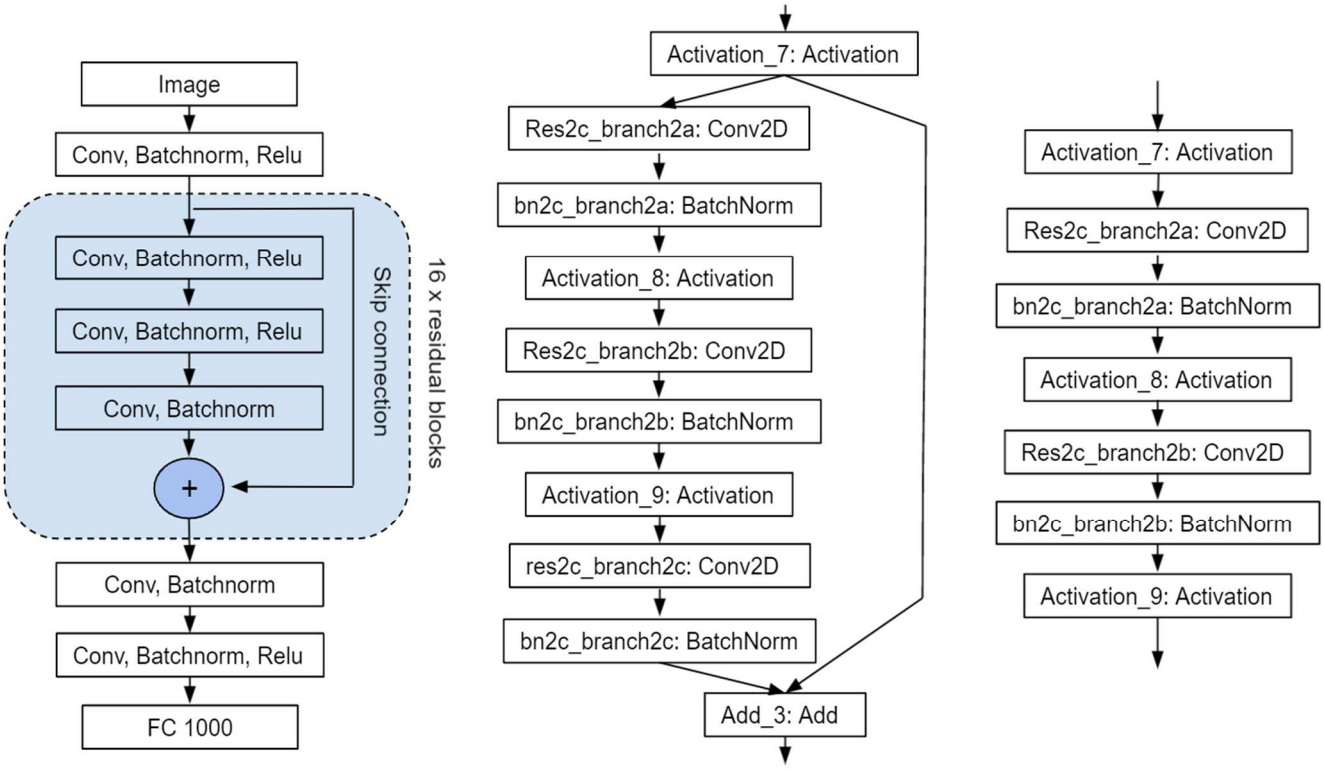


Fig. 4 Number of parameters from 49 pruning points

The ReLU layer chosen, lied in every residual block in Resnet50. There are 16 residual blocks in the Resnet50 network. Each residual block contains about 2 ReLU layers, 3 Convolutional (Conv) layers, and 2 Batch Normalization (BN) layers. A residual block also has a skip connection that sums the input and output layer of the residual block. This makes the Resnet network solves the vanishing gradient problem on a very deep network [8]. The pruning makes the skip connection disconnected at that residual layer. Fig. 5(a), 5(b), and 5(c) show the illustration of the process.



(a) (b) (c)

Fig. 5 The pruning illustration (a) A simplified Resnet50 architecture (b) An example of a Resnet50 residual block (c) The pruning result on a residual block

We simplify Resnet50 architecture in Fig. 5(a) because of its large size. In the real Resnet50, some residual blocks do not use skip-connection directly but use the Conv and BN layer, but we do not show them in simplified architecture.
We take one residual block, for example, which is the third residual block in Resnet50 (Fig. 5(b)). The third residual block starts with Activation_7 and add_3 at the end. It consists of eight layers that are 3 Conv2D layers, 3 BatchNorm layers, and 2 Activation layers. If we prune this residual block at Activation_9 layer, we will get the result that the residual block will be opened and become a straight layer (Fig 5(c)). We lost one residual block, but we still have some residual blocks at the previous layer.

The pruned network needs more layers to be able to classify data. Usually, an FC layer adds to the end of the network as a classifier and makes it suitable for the number of classes of the new dataset. From Fig. 5(c), activation_9 is a ReLU layer with a flattened output size is 193600. This layer will be flattened before entering the FC layer and produce a large number of parameters because it will be multiplied with the output FC layer directly.

For reducing this value, a Conv 1x1 kernel layer was added before the FC layer. A 1x1 Conv layer can downsampling the input feature map [7]. The formula to calculate the number of parameters in a Conv or FC layer is:

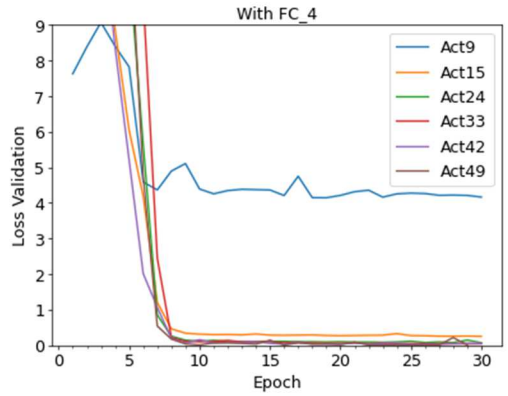
$$N_p = \left(k1 * k2 * N_{in} * N_{out} + N_{bias\_out}\right) \quad (1)$$

For the Conv layer, *k1* and *k2* are the kernel size, and for the FC layer, these values for both were 1 due to flattening. For the Conv layer, N_{in} was the number of feature maps (FM) at the previous layer, but for the FC layer, N_{in} was the flattened all input FM at the previous layer. If the size of FM were large enough, flattening would produce a large number of parameters. This is why fewer 1x1 Conv filters will reduce the flattened feature map. For the Conv layer, N_{out} was the number of FM at the output, and for the FC layer, N_{out} was the number of the output. N_{bias_ouput} was the number of the bias at the output for both layers.

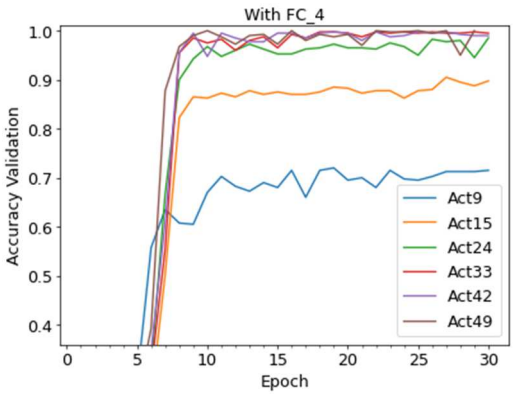
TABLE I
THE THREE TYPES OF NETWORKS TESTED AND THE NUMBER OF PARAMETERS FOR EACH PRUNING POINT

No		Number of Filter for downsampling	Flatten and Classifier	Number of parameters (x10^6)					
				Act9	Act15	Act24	Act33	Act42	Act49
1	Pruned network	-	Flatten + FC_4	0.99	1.23	2.38	6.28	11.58	23.9
2	Pruned network	32 Filters	Flatten + FC_4	0.60	0.93	2.22	6.11	11.50	23.7
3	Pruned network	64 Filters	Flatten + FC_4	0.99	1.04	2.25	6.14	11.52	23.7

Table I shows the model created and the number of parameters for 6-pruning points (Act9, Act15, Act24, Act33, Act42, Act49). It describes that pruning can reduce the number of parameters from more than 23 million to less than one million. This research uses two filters for downsampling the FM. We choose the number of filters close to the four classes of tobacco leaf pest. We decided 32 and 64 filters that considered can represent another number of filters. Thus, there are three types of architecture to be tested: without a filter, 32 filters, and 64 filters for downsampling. Table I shows that using 32 filters will have the least number of parameters, followed by 64 filters, and without the filter. After the filter, we add a flatten layer to change the dimension of the feature map to be 1x1 and the depth size according to the size and number of FM in the previous filter. The flatten layer makes sure the output is suitable for the FC layer. FC_4 was the fully connected layer with 4-outputs to handle 4-classes.



(a)



(b)

Fig. 6 The validation loss (a) and accuracy (b) using FC_4

III. RESULT AND DISCUSSION

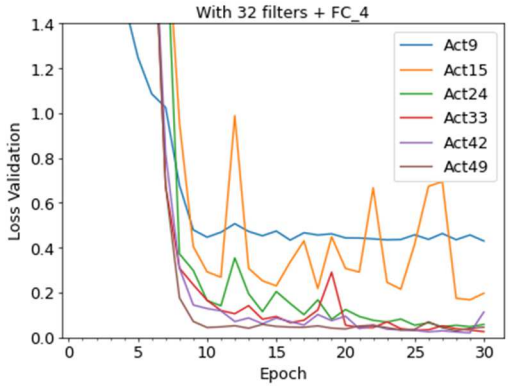
A. Training Results

We use the Keras framework (TensorFlow as backend) with 30 epochs, 0.0001 for learning rate, stochastic gradient descent for the optimizer, and categorical cross-entropy for loss calculation.

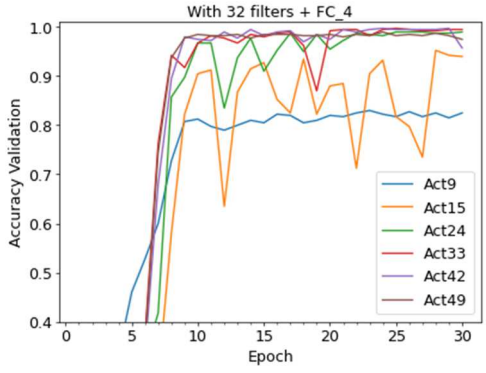
TABLE II
THE RESULT OF THREE TYPES NETWORKS TESTED ON VALIDATION DATASET

Pruning point	Min validation loss (%)			Max validation accuracy (%)			Reduction rate (%)		
	-	32 filters	64 filters	-	32 filters	64 filters	-	32 filters	64 filters
Act9	414.16	42.91	37.61	72	83	84.75	95.89	97.49	95.87
Act15	24.45	16.67	14.43	90.5	95.25	95.5	94.88	96.12	95.68
Act24	6.45	4.66	4.6	98.5	99	99.3	90.06	90.76	90.62
Act33	3.43	2.47	2.35	99.75	99.75	100	73.83	74.52	74.38
Act42	2.05	1.93	1.33	99.75	99.75	100	51.74	52.06	51.96
Act49	0.19	2.83	2.16	100	99	99.25	0.00	1.37	1.07
Average	75.12	11.91	10.48	93.41	95.96	96.42			

Fig. 6, 7, and 8 present the training results graph for the validation data on 6-pruning points. We expose only the validation data, not the training data, because validation data generally show the success of the training. Usually, using a training dataset will get better results than validation data during the training process for loss and accuracy. The network updates its weight based on the training dataset, not the validation dataset. The validation dataset is used to check if the training can adapt to the variation of the dataset well or not.



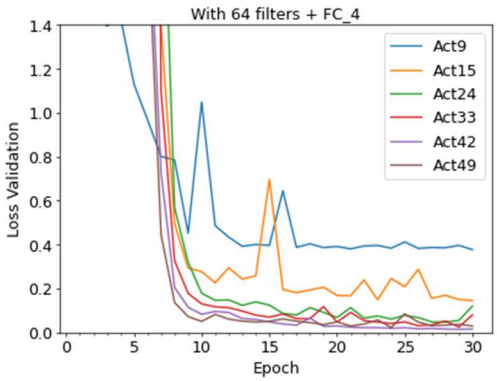
(a)



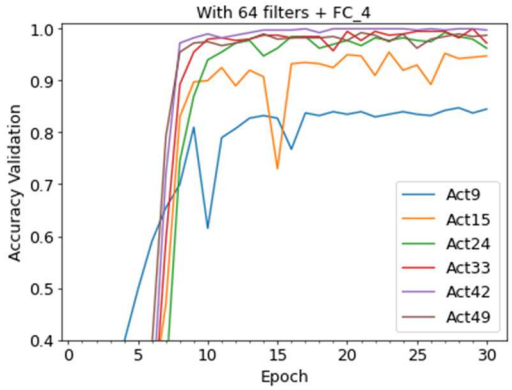
(b)

Fig. 7 The validation loss (a) and accuracy (b) using 32-filters and FC_4

Fig. 6, 7, and 8 show that the training process can converge. The loss validation value (at Fig. 6(a), 7(a), and 8(a)) decreases as the epoch increases. In the early epochs, the validation loss decreased fast, and after about the 10th epoch, the decrease in validation loss started to slow down and stabilize. Validation accuracy also increases rapidly in the early epoch (at Fig. 6(b), 7(b), and 8(b)). It also needs around the 10th epoch upwards to start to be a steady slope. Transfer learning makes it convergence fast for just a few epochs applied.



(a)



(b)

Fig. 8 The validation loss (a) and accuracy (b) using 64-filters and FC_4

It looks that the fluctuation value of loss and validation in addition to downsampling filter (32 and 64 filters) was higher than without downsampling filter. This is because filters make more layers added with random weight. It is fine, but it usually needs some extra epoch to update the weights and make it more stable.

A more detailed comparison of loss and accuracy validation values can be seen in Table II. We choose minimum for loss value and maximum for accuracy because they represent the best result during training. We also show the reduction rate in the tables to show how many parameters was removed for each pruning point.

We can see that the average value of loss validation for three network architectures is 75.12, 11.91, and 10.48. This explains that the addition of 64 filters is better than the others according to the average loss validation value. The average

accuracy values are 93.41, 95.96, and 96.42, respectively. Also, it shows that the addition of 64 filters is better than the others according to the average validation accuracy value.

We are going to see how the pruning maintains the accuracy. Firstly, for the addition just of FC_4 with pruning point at activation-24, we can see that pruning reduces the number of parameters up to 90.06% (from an almost not pruning network) can maintain accuracy up to 98.5%. For the addition of 32 and 64 filters, the reduction rates are 90.76% and 90.62%, maintaining up to 99.0 and 99.3%, respectively. The addition of 64 filters can maintain accuracy up to 100% for the reduction rate of 74.38% at the pruning point of activation-33.

The best result was the network that can maintain accuracy up to 100% like the unpruned network. It could be reached using the pruning point at Act33 (activation-33) layer and 64 filters for dimensionality reduction. In this case, the reduction rate was 74.38%, which means it reduced from 23.7 million to 6.14 million in the number of parameters (see Table I). We also got a higher reduction rate for some pruning points, but the accuracy decreases slightly (99.0% and 99.3%), for example, at activation-24 pruning points with 32 and 64 filters.

IV. Conclusion

The implementation of the depth pruning of Resnet50 could work pretty well in the tobacco leaf pest dataset. The depth pruning often produces many feature maps at the end of the network. They need to be reduced to decrease the number of parameters. We applied a 1x1 kernel convolutional layer as a downsampling or dimensional reduction for the number of feature maps. We used the validation dataset to show the performance. We got the best performance as an unpruned network using pruning point at Activation-33 layer and 64 filters for downsampling. At the best performance, we accuracy up to 100% like the unpruned network. It reduces the number of parameters from 23.7 million (unpruned network) reduced up to 6.14 million, or about a 74.38% reduction rate.

Acknowledgment

The tobacco leaf data from this study was supported by PT Perkebunan Nusantara X (PTPN X) Jember, East Java. PTPN X is a state-owned enterprise with the main business, namely the sugar and tobacco industry.

References

[1] C. S. Marzan and C. R. Ruiz, "Automated tobacco grading using image processing techniques and a convolutional neural network," Int. J. Mach. Learn. Comput., vol. 9, no. 6, pp. 807–813, 2019, doi: 10.18178/ijmlc.2019.9.6.877.

[2] A. Harjoko, A. Prahara, T. W. Supardi, I. Candradewi, R. Pulungan, and S. Hartati, "Image processing approach for grading tobacco leaf based on color and quality," Int. J. Smart Sens. Intell. Syst., vol. 12, no. 1, pp. 1–10, 2019, doi: 10.21307/ijssis-2019-010.

[3] D. S. Guru, P. B. Mallikarjuna, S. Manjunath, and M. M. Shenoi, "Machine Vision Based Classification Of Tobacco Leaves For Automatic Harvesting," Intell. Autom. Soft Comput., vol. 18, no. 5, pp. 581–590, 2012, doi: 10.1080/10798587.2012.10643267.

[4] Y. Sun, H. Q. Wang, Z. Y. Xia, J. H. Ma, and M. Z. Lv, "Tobacco-disease Image Recognition via Multiple-Attention Classification Network," J. Phys. Conf. Ser., vol. 1584, no. 1, 2020, doi: 10.1088/1742-6596/1584/1/012008.

[5] D. I. Swasono, H. Tjandrasa, and C. Fatichah, "Classification of tobacco leaf pests using VGG16 transfer learning," Proc. 2019 Int. Conf. Inf. Commun. Technol. Syst. ICTS 2019, pp. 176–181, 2019, doi: 10.1109/ICTS.2019.8850946.

[6] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, "Gradient-based learning applied to document recognition", Proc. IEEE 86 (11) (1998) 2278–2324, 1998.

[7] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, "Going deeper with convolutions", In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1–9, 2015.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition", IEEE Conf. Comput. Vis. Pattern Recognit. 770–778 (2016). doi:10.1109/CVPR.2016.90, 2016.

[9] J. Zou, T. Rui, Y. Zhou, C. Yang, and S. Zhang, "Convolutional neural network simplification via feature map pruning," *Comput. Electr. Eng.*, vol. 70, pp. 950–958, 2018, doi: 10.1016/j.compeleceng.2018.01.036.

[10] G. Li, J. Wang, H. W. Shen, K. Chen, G. Shan, and Z. Lu, "CNNPruner: Pruning convolutional neural networks with visual analytics," *IEEE Trans. Vis. Comput. Graph.*, vol. 27, no. 2, pp. 1364–1373, 2021, doi: 10.1109/TVCG.2020.3030461.

[11] B. O. Ayinde, T. Inanc, and J. M. Zurada, "Redundant feature pruning for accelerated inference in deep neural networks," *Neural Networks*, vol. 118, pp. 148–158, 2019, doi: 10.1016/j.neunet.2019.04.021.

[12] C. Yang *et al.*, "Structured Pruning of Convolutional Neural Networks via L1 Regularization," *IEEE Access*, vol. 7, pp. 106385–106394, 2019, doi: 10.1109/ACCESS.2019.2933032.

[13] P. Singh, V. K. Verma, P. Rai, and V. P. Namboodiri, "Acceleration of Deep Convolutional Neural Networks Using Adaptive Filter Pruning," IEEE J. Sel. Top. Signal Process., vol. 14, no. 4, pp. 838–847, 2020, doi: 10.1109/JSTSP.2020.2992390.

[14] C. Liu and H. Wu, "Channel pruning based on mean gradient for accelerating Convolutional Neural Networks," *Signal Processing*, vol. 156, pp. 84–91, 2019, doi: 10.1016/j.sigpro.2018.10.019.

[15] A. H. Ashouri, T. S. Abdelrahman, and A. Dos Remedios, "Retraining-free methods for fast on-the-fly pruning of convolutional neural networks," *Neurocomputing*, vol. 370, no. xxxx, pp. 56–69, 2019, doi: 10.1016/j.neucom.2019.08.063.

[16] F. E. Fernandes and G. G. Yen, "Pruning Deep Convolutional Neural Networks Architectures with Evolution Strategy," *Inf. Sci. (Ny).*, vol. 552, pp. 29–47, 2021, doi: 10.1016/j.ins.2020.11.009.

[17] S. K. Yeom *et al.*, "Pruning by explaining: A novel criterion for deep neural network pruning," *Pattern Recognit.*, vol. 115, 2021, doi: 10.1016/j.patcog.2021.107899.

[18] A. Jordao, M. Lie, and W. R. Schwartz, "Discriminative Layer Pruning for Convolutional Neural Networks," *IEEE J. Sel. Top. Signal Process.*, vol. 14, no. 4, pp. 828–837, 2020, doi: 10.1109/JSTSP.2020.2975987.

[19] F. Tung and G. Mori, "Deep Neural Network Compression by In-Parallel Pruning-Quantization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 3, pp. 568–579, 2020, doi: 10.1109/TPAMI.2018.2886192.

[20] Y. Liang, W. Liu, S. Yi, H. Yang, and Z. He, "Filter pruning-based two-step feature map reconstruction," *Signal, Image Video Process.*, vol. 15, no. 7, pp. 1555–1563, 2021, doi: 10.1007/s11760-021-01888-4.

[21] C. Qi *et al.*, "An efficient pruning scheme of deep neural networks for Internet of Things applications," *EURASIP J. Adv. Signal Process.*, vol. 2021, no. 1, 2021, doi: 10.1186/s13634-021-00744-4.

[22] E. Jeczmionek and P. A. Kowalski, "Flattening layer pruning in convolutional neural networks," *Symmetry (*Basel*).*, vol. 13, no. 7, pp. 1–13, 2021, doi: 10.3390/sym13071147.

[23] S. Zhang, G. Wu, J. Gu, and J. Han, "Pruning convolutional neural networks with an attention mechanism for remote sensing image classification," Electron., vol. 9, no. 8, pp. 1–19, 2020, doi: 10.3390/electronics9081209.