Comparative Performance of Various Deep Learning based Models in Fruit Image Classification

Raheel Siddiqi
Computer Science Department
Bahria University
Karachi, Pakistan
drraheel.bukc@bahria.edu.pk

ABSTRACT

Fruit type classification based on computer vision remains challenging because of shape, color and texture similarity among numerous species of fruits. Recent research results indicate that the Convolutional Neural Network based models have demonstrated substantial improvements in fruit image classification accuracy. This paper proposes fourteen different deep learning models for fruit type classification. Thirteen of these models exploit ImageNet pre-trained feature extractors. In addition to these thirteen models, a simple, lightweight 12-layer model called *FruitNet* has also been proposed. A large, publicly available fruit images dataset, called Fruits 360, has been used to train and evaluate the proposed models. Among all the models that exploited some pre-trained feature extractor, Xception based model has produced the best validation accuracy rate (i.e. 99.97%) while MobileNet based model has proved to be a very efficient model. Fine tuning has also been found to produce excellent results. In fact, the best validation accuracy of 99.98% is achieved when VGG16 based model is fine tuned. On the other hand, FruitNet has yielded accuracy rate of 99.73% which is excellent given its small size and simple structure. FruitNet is also the most efficient model among all the proposed models.

CCS CONCEPTS

• Computing methodologies → Object recognition • Computer systems organization → Neural networks.

KEYWORDS

Fruit Image Classification, Transfer Learning, Fine Tuning, Convolutional Neural Networks, Computer Vision

ACM Reference format:

Raheel Siddiqi. 2020. Comparative Performance of Various Deep Learning based Models in Fruit Image Classification. In *Proceedings of International Conference on Advances in Information Technology (IAIT2020), July 1-3, 2020, Bangkok, Thailand. ACM, New York, NY, USA, 9 pages.* https://doi.org/10.1145/3406601.3406619

IAIT2020, July 1–3, 2020, Bangkok, Thailand

ACM ISBN 978-1-4503-7759-1/20/07…$15.00
https://doi.org/10.1145/3406601.3406619

1 Introduction

Image-based fruit type classification is considered as a difficult problem. The reason is that fruits are of a variety of different size, texture, color and shape. A number of applications are possible for a reliable fruit classification system. Such a system may be exploited at POS terminals of supermarkets and can help cashiers in determining fruit type as well as its variety [1]. This will enable accurate fruit price calculation. Another possible application is in smart phone based software that provides dietary advice to people (especially those who have health issues). The application will identify the fruit species and its variety as well as display its nutritional content and its suitability for the person (based upon his/her health conditions) [2]. Costly techniques and machinery like near-infrared imaging [3], gas sensors [4] and electronic nose [5] have been tried for fruit detection but they require professional operators and also suffer from low accuracy rates [6]. In addition, supermarkets cannot afford such expensive equipment [7]. Therefore, researchers are nowadays focusing on computer vision techniques for fruit image classification. No expensive hardware is required in the case of computer vision techniques as input is taken through digital cameras which are much less expensive.

Earlier computer vision based fruit image classification systems [1, 7, 8, 9, 10, 11] exploited shallow learning i.e. they all relied on hand-crafted features. Recently, researchers are focusing on deep learning based solutions [2, 6, 12, 33] for fruit image classification. Deep learning methods are known for automatic feature engineering [2, 13] i.e. discriminative features are automatically learned by the model during training. This paper proposes fourteen different deep learning based models for image based fruit type classification. Thirteen of these models exploit feature extractor (i.e. the convolutional base) of an ImageNet [14] pre-trained model [15, 16, 17, 18, 19, 20, 21, 22]. Feature extractors of the following pre-trained models have been used:

- VGG16 and VGG19 [16],
- Xception [15],
- ResNet50 [17],
- InceptionV3 [18],
- InceptionResNetV2 [19],
- MobileNet [20],
- DenseNet121, DenseNet169 and DenseNet201 [21], and
- MobileNetV2 [22]

In addition to the thirteen models mentioned above, a simple, lightweight 12-layer model called FruitNet has also been proposed in this paper. FruitNet is totally designed by the author himself and no transfer learning is used in this case. A large, publicly available dataset of fruit images called Fruits-360 dataset [24] is exploited for the training and evaluation of all models. The rest of this paper is organized as follows. Section 2 presents review of recent related works. Section 3 presents a brief overview of the Fruits-360 dataset. Section 4 describes the methods and the experimental setup. Section 5 presents and analyses the results of the experiments. Section 6 concludes the paper by summarizing the key findings of the research. Section 6 also indicates directions for future work.

2 Related Works

Image-based Fruit type classification systems developed in the shallow learning era [1, 7, 8, 9, 10, 11] had two weaknesses [6]: (1) they all used handcrafted features, and (2) the classifiers were of simpler structures and therefore lacked the ability to map the complicated features to the final classification result. On the other hand, deep learning methods (such as CNN) can help overcome these weaknesses. They normally require very less data preprocessing as they learn the features themselves. They are representation-learning methods with multiple levels of representation for the input image. Each level of representation is more abstract than the previous one. For image classification tasks, higher levels of representation enhance the discriminative features and suppress irrelevant variations [13]. This enables classification systems to map intricate differences to the final classification result.

Zhang et al. [6] constructed a 13-layer deep CNN for fruit image classification. Out of these 13 layers, only 6 layers had learnable weights and biases. The dataset size (used in the study) was 3600 with 18 fruit types and 200 images per fruit type. Data preprocessing involved four steps: (1) the fruit was moved to the center of each image, (2) the image was resized to a 256x256 matrix, (3) background was removed, and (4) each image was labeled manually to one of the 18 fruit types. The training data was increased in size through various data augmentation techniques. The model's accuracy was 94.94% which was much higher than the previous shallow learning based systems [1, 7, 8, 9, 10, 11].

Wang and Chen [12] tried to improve the work presented in [6]. They constructed an improved 8-layer deep CNN. Actually, the overall CNN size was 17 layers but only 8 layers had learnable weights and biases. Two major improvements were: (1) parametric rectified linear units were used instead of plain rectified linear units, and (2) dropout layer was placed before each fully connected layer to help training fully connected layers. The same dataset of [6] was used in [12]. Through data augmentation, the training set size was increased from 1800 images to 325,800 images. The test set classification accuracy, reported in [12], was 95.67%. This was 0.73% better than that of the CNN used in [6].

Siddiqi [33] has recently experimented with both InceptionV3 and VGG16 pre-trained models and has reported high accuracy rates for the Fruits 360 dataset [24] containing 72 classes[1]. The highest classification accuracy rate reported in the study is 99.27% and this is achieved using VGG16 model. The results presented in [2, 33] form the basis of the underlying motivation to carry out the study presented in this paper. Fourteen different deep learning models are proposed and systematically evaluated and analyzed in this paper.

3 Fruits 360 Dataset

Fruits 360 dataset consists of 69,802 fruit images (52262 images are present in the training set while the validation set has 17540 images) [23], [24]. There are 101 different fruit classes in the dataset and each image contains only one fruit. The images are obtained by making a short twenty seconds video of the fruit while it is slowly rotated by a motor and then extracting frames/images from that video [23]. A Logitech C920 camera is used to record videos. After the images have been extracted from the video, each image's background is removed through a dedicated algorithm. This is done because background is not uniform due to varying light conditions. In most cases around 490 training images and 164 validation images are present for each fruit type. However, in some cases the number of images in the training and validation sets varies slightly. Each image in the dataset is of 100 X 100 pixels. Figure 1 depicts some images from the Fruits 360 dataset.

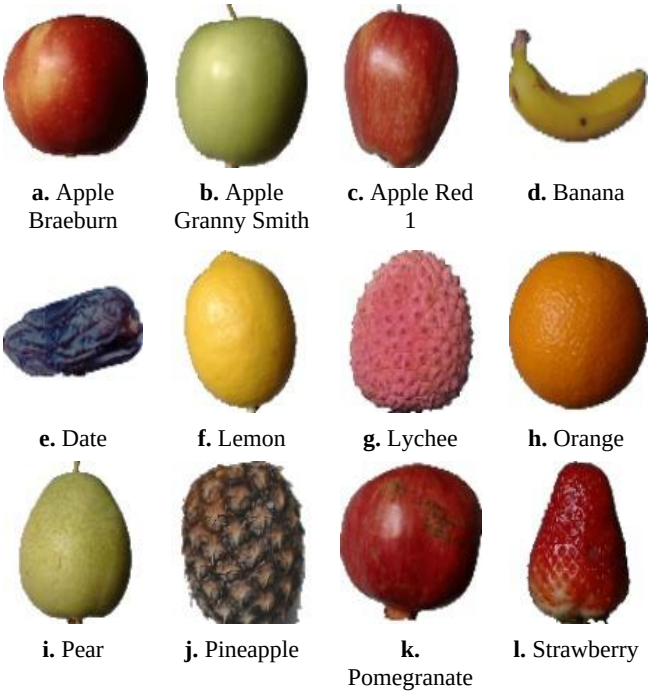

Figure 1. Some images from Fruits 360 dataset. Class label for each image is also given.

[1] Fruits 360 dataset consisted of 72 classes at that time. More classes were added later.

4 Methods

In this section, some deep-learning based computer vision techniques and models are briefly described. The techniques and models, presented in this section, form the basis for the experiments presented later in this paper.

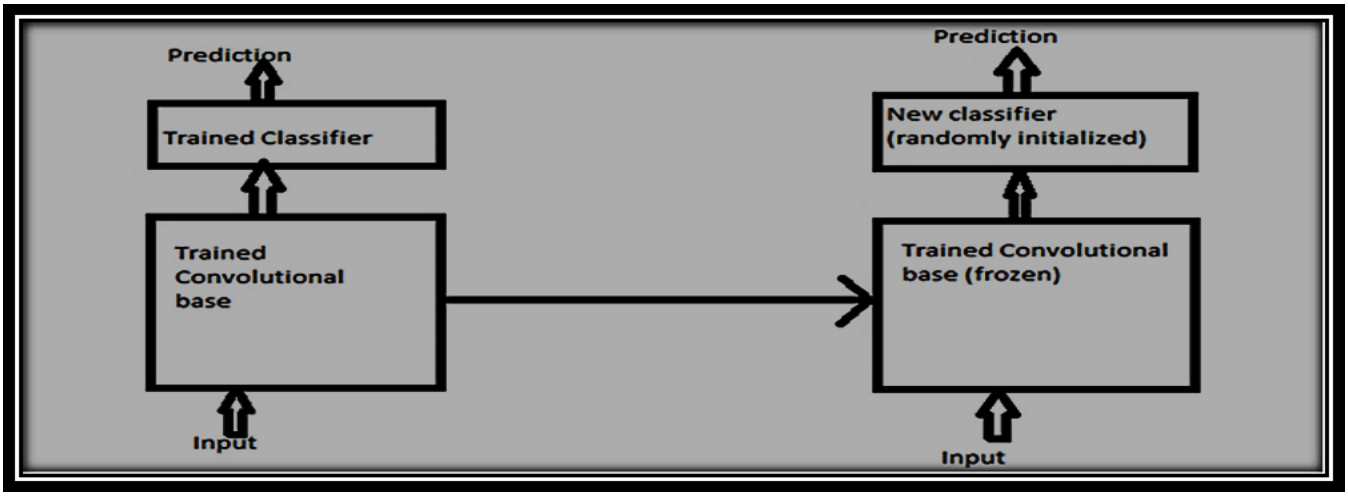

Figure 2. Transfer learning involves changing classifier while keeping the same convolutional base.

4.1 FruitNet

FruitNet is a simple, lightweight 12-layer CNN designed by the author to classify the Fruits 360 dataset images. FruitNet architectural design details are presented in Table 1. Unlike other models proposed in this paper, transfer learning is not exploited in FruitNet. Only six layers have trainable weights and biases i.e. four convolutional layers and two fully-connected dense layers. FruitNet has been specially designed to overcome overfitting. This is done by incorporating a number of regularization techniques. The following regularization techniques are implemented in the FruitNet design:

- **Use of a low capacity network.** The simplest way to prevent overfitting is to use a smaller network i.e. less number of layers and less number of units per layer. This is the reason why FruitNet has only four convolutional layers and only two dense layers. The number of filters/units in each of these layers is also kept to a bare minimum. While reducing the size of the network it was also ensured that the model does not underfit.
- **Dropout layers have been added.** Three dropout layers are included in FruitNet in order to make sure that the model does not learn insignificant patterns in the train set.
- **Weight regularization has been added for the first dense layer.** The first dense layer has a very large number of parameters/weights. The weights are regularized using L2 regularization. This will force weights to take only small values and therefore simplifies the FruitNet model.

In addition to these details, it is also worthwhile mentioning that each convolutional layer uses a kernel size of 3x3 and each max-pooling layer uses a pool size of 2x2. The first eight layers are responsible for feature extraction, while the last four layers perform final classification. Since no transfer learning is used, FruitNet is going to be trained from scratch i.e. all the model weights are initially randomly initialized.

Table 1. Twelve layers of the FruitNet model

Layer	*Purpose*	*Output Shape*
1	Convolution+ReLU	(Batch Size, 98, 98, 32)
2	Convolution+ReLU	(Batch Size, 96, 96, 32)
3	Max-Pooling	(Batch Size, 48, 48, 32)
4	Dropout (rate=0.2)	(Batch Size, 48, 48, 32)
5	Convolution+ReLU	(Batch Size, 46, 46, 64)
6	Convolution+ReLU	(Batch Size, 44, 44, 64)
7	Max-Pooling	(Batch Size, 22, 22, 64)
8	Dropout (rate=0.2)	(Batch Size, 22, 22, 64)
9	Flatten	(Batch Size, 30976)
10	Dense (units=512)	(Batch Size, 512)
11	Dropout (rate=0.2)	(Batch Size, 512)
12	Dense (units=101)	(Batch Size, 101)

4.2 Transfer Learning and Fine Tuning

Transfer learning is the process of reusing a pre-trained model trained on a large dataset, typically on a large-scale image classification task [25, 26]. A model may be trained on ImageNet [14] (where classes are mostly animals and everyday objects) and then the features learned may be utilized for a task like fruit image classification. So, for example, VGG16 model (like all other pre-trained models considered in this study) has been trained on ImageNet dataset. The model's network comprise of

two parts: (1) a convolutional base (i.e. the feature extractor) which is a series of convolution and pooling layers, and (2) a densely connected classifier. During transfer learning, the convolutional base is retained but the trained classifier is removed and a new classifier is added which is randomly initialized. This new classifier is then trained on the output of the convolutional base so that it can identify new classes (e.g. classes of the fruit image dataset). Figure 2 depicts the process of transfer learning.

Fine tuning is another widely used technique for model reuse. In fine-tuning, we optimize both the weights of the newly added classifier as well as the weights of some (or all) of the layers of the convolutional base. Figure 3 depicts an example of fine tuning applied on the VGG16 model [16]. It must be remembered that fine-tuning the top layers of the convolutional base is only possible once the newly added classifier on top has already been trained. If the classifier is not already trained, the error signal propagated backwards during training will be too large and will destroy the abstract representations previously learned by the convolutional base's top layers that are being fine-tuned [25]. Due to this reason, the recommended steps for fine-tuning are [25]:

1. Add a custom classifier on top of the network's convolutional base. All the weights of the convolutional base should remain frozen.
2. Train the custom classifier.
3. Unfreeze some layers in the convolutional base
4. Jointly train both the unfrozen layers and the custom classifier.

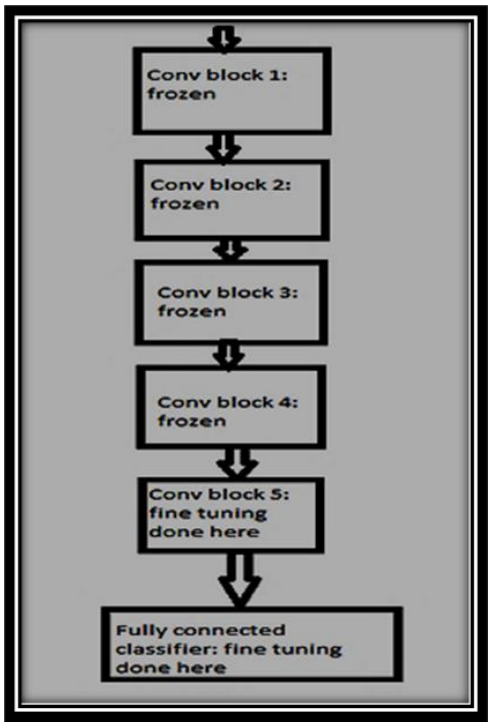

Figure 3. Fine tuning the last convolutional block and the fully connected classifier.

All the proposed models presented in this paper (except FruitNet) are based on the concepts of transfer learning and fine tuning. These two concepts enable the exploitation of the pre-trained models [15, 16, 17, 18, 19, 20, 21, 22] for image based fruit type classification.

4.3 Pre-Trained Models

Most proposed models, presented in this study, exploit convolutional base of one of the ImageNet pre-trained models [15, 16, 17, 18, 19, 20, 21, 22]. Table 2 summarizes key attributes of the various pre-trained models used in the study.

Table 2. Summary of the key attributes of the various pre-trained models. The top-1 and top-5 accuracies refer to the models' performance on the ImageNet's validation dataset. Depth refers to the total number of layers, including both trainable and non-trainable layers.

Model	*Size*	*Top-1 Accuracy*	*Top-5 Accuracy*	*Depth*
VGG16	528 MB	0.713	0.901	23
VGG19	549 MB	0.713	0.900	26
Xception	88 MB	0.790	0.945	126
ResNet50	98 MB	0.749	0.921	-
InceptionV3	92 MB	0.779	0.937	159
InceptionResNetV2	215 MB	0.803	0.953	572
MobileNet	16 MB	0.704	0.895	88
DenseNet121	33 MB	0.750	0.923	121
DenseNet169	57 MB	0.762	0.932	169
DenseNet201	80 MB	0.773	0.936	201
MobileNetV2	14 MB	0.713	0.901	88

4.4 Experimental Setup

All the experiments are carried out using TensorFlow version 1.13.1 and Keras version 2.2.4. All the code is written and executed on Jupyter notebooks[2]. Table 3 gives hardware specification for the experiments.

Table 3. Hardware specification for the experiments

CPU	Intel® Core™ i7 7700HQ
RAM	16 GB
GPU	NVIDIA's GeForce® GTX 1050 Ti

Figure 4 presents the network architecture that has been used in all the models where transfer learning or fine tuning is used i.e. all models except FruitNet. The network architecture consists of two main parts: (1) the convolutional base, and (2) the classifier. One of the pre-trained models (e.g. VGG16, VGG19, Xception, ResNet50 etc.) is used as the convolutional base[3]. The classifier consists of a flatten layer and two dense layers. The first dense layer has 512 nodes and uses 'ReLU' as the activation function.

[2] Github repository for all the experiments (presented in this paper): https://github.com/raheelsiddiqi2013/fruit_image_classification_101_classes

[3] It is important to remember that only the convolutional base of a particular pre-trained model is reused. The trained classifier of the pre-trained model is removed and replaced by a custom classifier.

The second dense layer has 101 nodes (corresponding to 101 fruit classes) and uses 'softmax' as the activation function. On the other hand, the network architecture for FruitNet is given in section 4.1.

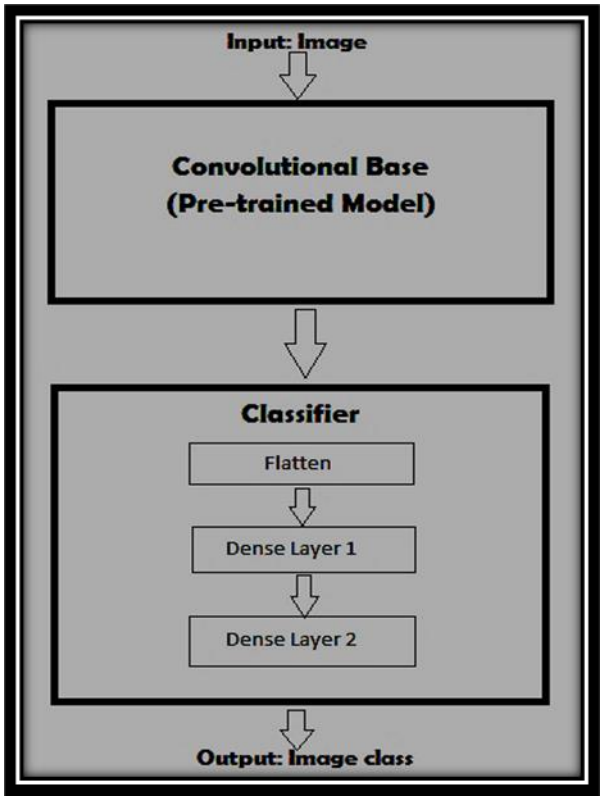

Figure 4. Network architecture used by the models based on transfer learning or fine tuning

Table 4 presents the settings for the training hyper parameters. These settings are used for all the models (including FruitNet). Since the networks are designed to solve a multi-class classification problem, 'categorical cross entropy' is used as the loss function. The loss function is the objective that every model tries to minimize during training. 'Adam' is chosen as the optimizer because it is a simple and computationally efficient technique for gradient-based optimization of loss functions [28]. Empirical results demonstrated in [28] indicate that 'Adam' works better than other stochastic optimization methods. A learning rate of 1e-4 is used because it is deemed to be neither too high nor too low. Unlike other proposed models, all the weights of FruitNet are initially randomly initialized. Therefore, FruitNet has to be trained from scratch. For other proposed models, the convolutional base (or the feature extractor) is already trained. This is the reason why FruitNet is trained for a greater number of epochs compared with the other proposed models.

For the experiments where fine tuning is also performed, there are some changes in the training process as well as the hyper parameter settings. It is important to mention here that fine tuning is only done for the VGG16 and VGG19 based models and not for all the models considered in this study. If an experiment involves fine tuning then the model is first trained for 30 epochs with the convolutional base frozen i.e. conventional transfer learning is performed for the first 30 epochs. The learning rate is kept at 1e-4. After 30 epochs, all the layers of the model's last convolutional block are unfrozen (see Figure 3 and Section 4.2) and the network is trained for another 30 epochs with a lower learning rate of 1e-5. Rest of the hyper parameter settings remain the same as Table 4.

Table 4. Hyper parameter settings for the experiments

Loss function	Categorical Cross-Entropy
Optimizer	Adam
Learning rate	1e-4
Number of epochs	50 (for all models except FruitNet), 75 (FruitNet)
Steps per epoch	1634 (Number of training samples/batch size)
Batch size	32

Data augmentation has been shown to be effective in image classification [29]. The competition wining classifiers of [16, 17, 27, 30] have all utilized data augmentation. Data augmentation allows researchers to artificially inflate training data and therefore helps in preventing overfitting [31]. Table 5 presents Keras [32] based data augmentation settings for the experiments. Keras' `ImageDataGenerator` class has been exploited to perform real-time data augmentation. Apart from data augmentation, the image data is also normalized by scaling pixel values to the range 0-1.

Table 5. Keras based data augmentation settings for the experiments

Rotation Range	40 degrees
Width Shift Range	0.2
Height Shift Range	0.2
Shear Range	0.2
Zoom Range	0.2
Horizontal Flip	True

5 Results and Analysis

Figure 5, Figure 6 and Figure 7 presents training curves for FruitNet and two other proposed models[4]. These three models have been selected to give an overall idea of the training process. Since FruitNet is trained from scratch, its training and validation losses started with relatively higher values and then gradually declined to lower values and then stabilized for the remaining epochs. This is in contrast to the Xception and DenseNet169 based models which both started with much lower loss values because they already had trained convolutional base.

In all the experiments, the model (i.e. a particular set of weight values) which gives the lowest validation loss (during training) is selected and is used to compute the final validation accuracy i.e.

[4] Training curves and evaluation results for all the proposed models are available at the following GitHub repository:
https://github.com/raheelsiddiqi2013/fruit_image_classification_101_classes

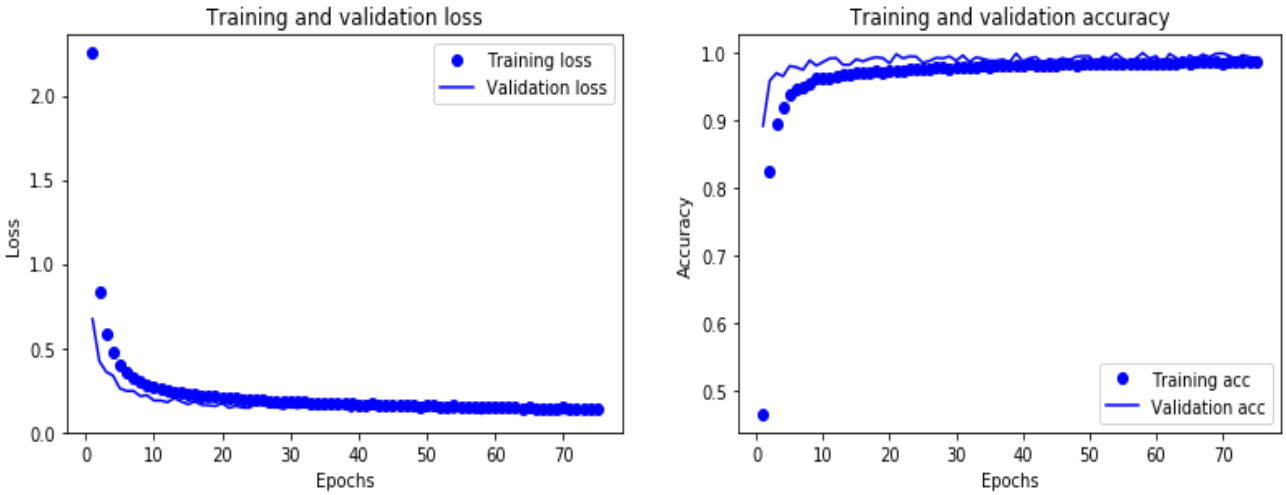

Figure 5. Training curves for the FruitNet model

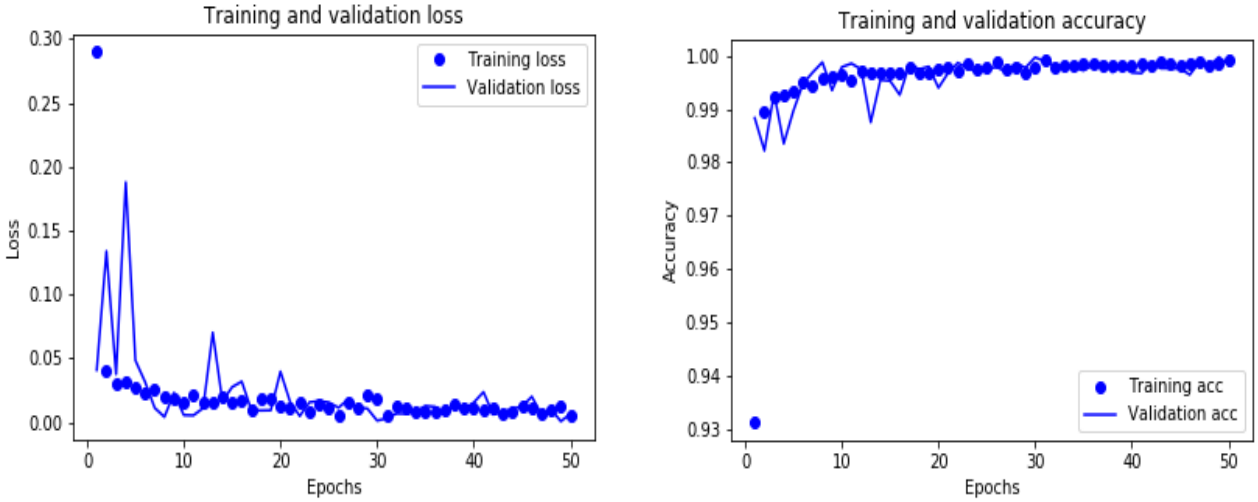

Figure 6. Training curves for the Xception based model

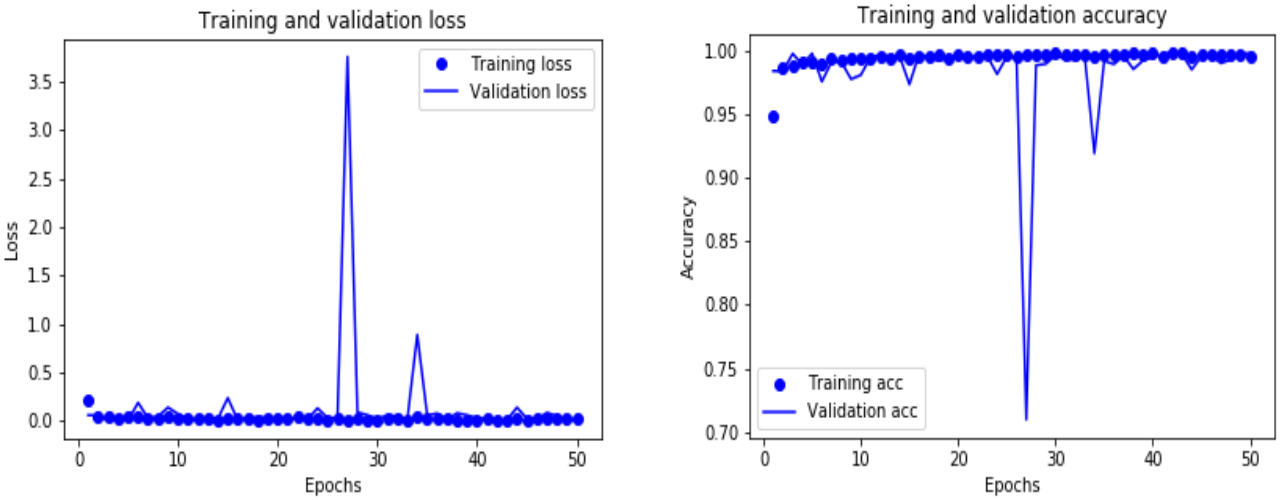

Figure 7. Training curves for the DenseNet169 based model

validation accuracies given in Table 6 and Table 7 are produced by the best models found for the respective CNN architectures. This is all based on the concept that lower the validation loss, the higher the validation accuracy. This phenomenon can also be observed in Figure 5, Figure 6 and Figure 7.

Table 6 and Figure 8 presents comparative performance of the various proposed deep learning based models in the context of image-based fruit type classification. As already stated, all the experiments are performed using the Fruits 360 dataset [24]. Table 6 demonstrates that excellent results are obtained for all the proposed models. Validation accuracy rates are quite high. The lowest accuracy rate recorded is 99.68%. InceptionResNetV2 and MobileNetV2 based models have yielded the lowest accuracy rate of 99.68%. On the other hand, Xception based model has yielded the best validation accuracy rate of 99.97%. These results are significant because the Fruits 360 dataset is a relatively large dataset consisting of 69,802 fruit images belonging to 101 different fruit classes. All the relevant recent research studies [1, 2, 6, 8, 9, 10, 11, 12] have exploited much smaller datasets. The best test-set classification accuracy reported in relevant literature is 99.75% on a dataset of 2633 fruit images [2]. Considering this, classification accuracy of 99.97% (produced by the Xception based model) is a significant improvement.

FruitNet, which is one of the proposed deep learning models, has also performed quite well. In terms of validation accuracy, it has performed better than InceptionResNetV2 and MobileNetV2 based models. It has produced a validation accuracy of 99.73% which is comparable to the accuracies of the other models. This performance is quite significant as FruitNet consists of a very simple, sequential architecture with only four convolutional layers. This is in stark contrast to the other proposed models well as the 'Validation Time'. It also has a pretty decent accuracy rate of 99.82%. Therefore, it is probably the best choice where high efficiency with reasonable accuracy is required (see Figure 8) which have much more complex feature extractors. Since FruitNet is lightweight, it is also the fastest model (see Table 6). It consumes the least amount of time during training as well as validation. taken to classify test images divided by the number of steps. After FruitNet, MobileNet based model has proved to be the second most efficient model, both in terms of the 'Train Time' as

The 'Train Time', given in Table 6, is equal to the minimum time taken to complete a training epoch divided by the number of steps per epoch. The 'Test Time' is equivalent to the total time. MobileNetV2 based model has the third best train and test times but at a relatively much lower accuracy rate. On the other hand, DenseNet201 based model has proved to be the slowest model both in terms of train and test times. Of the three DenseNet based models, DenseNet121 based model has the best performance both in terms of accuracy as well as speed.

Figure 8 visualizes the accuracy/speed trade-off of the proposed models. The models that appear on the top-left have the best accuracy and the best speed. On the other hand, the models that appear on the bottom-right have the worst accuracy and the worst speed. FruitNet as well as the models based on Xception, DenseNet121, VGG19 and InceptionV3 provide a decent balance between accuracy and speed. VGG19 based model has better accuracy than the one based on VGG16 but since VGG19 has more parameters, it is less efficient than the model based on VGG16 (see Table 2, Table 6 and Figure 8). InceptionV3 outperforms the InceptionResNetV2, especially in terms of speed. InceptionResNetV2 based model took much longer to train and much longer to test. In fact, InceptionResNetV2 based model was one of the worst performing models among all the models evaluated (see Figure 8). Model based on ResNet50 has produced mediocre performance.

Table 6. Comparative performance of the proposed models on the Fruits 360 dataset

Deep Learning Model	Validation accuracy	Train Time (per epoch)	Validation Time
VGG16 based model	99.75%	288ms/ step	86ms/step
VGG19 based model	99.89%	335ms/ step	97ms/step
Xception based model	99.97%	348ms/ step	83ms/step
ResNet50 based model	99.79%	349ms/ step	82ms/step
InceptionV3 based model	99.86%	267ms/ step	82ms/step
InceptionResNetV2 based model	99.68%	489ms/ step	127ms/step
MobileNet based model	99.82%	240ms/ step	70ms/step
DenseNet121 based model	99.95%	297ms/ step	98ms/step
DenseNet169 based model	99.93%	395ms/ step	112ms/step
DenseNet201 based model	99.86%	495ms/ step	133ms/step
MobileNetV2 based model	99.68%	263ms/ step	80ms/step
FruitNet	99.73%	236ms/ step	67ms/step

Table 7. Results obtained by fine tuning the VGG16 and VGG19 based models

Deep Learning Model	Validation accuracy	Train Time (per epoch)	Validation Time
VGG16 based model (fine tuned)	99.98%	292ms/step	89ms/step
VGG19 based model (fine tuned)	99.97%	349ms/step	102ms/step

Table 7 presents results obtained on fine tuning the VGG16 and VGG19 based models. For both the models, the layers of the last convolutional block (contained in the convolutional base) are unfrozen and jointly trained with the custom classifier that has been added on top of the convolutional base (see Figure 3). Accuracy rate of the VGG16 based model has increased to 99.98% due to fine tuning. Similarly, accuracy rate of VGG19 based model has increased to 99.97%. These are very high accuracy rates and almost approaching the perfect 100% classification

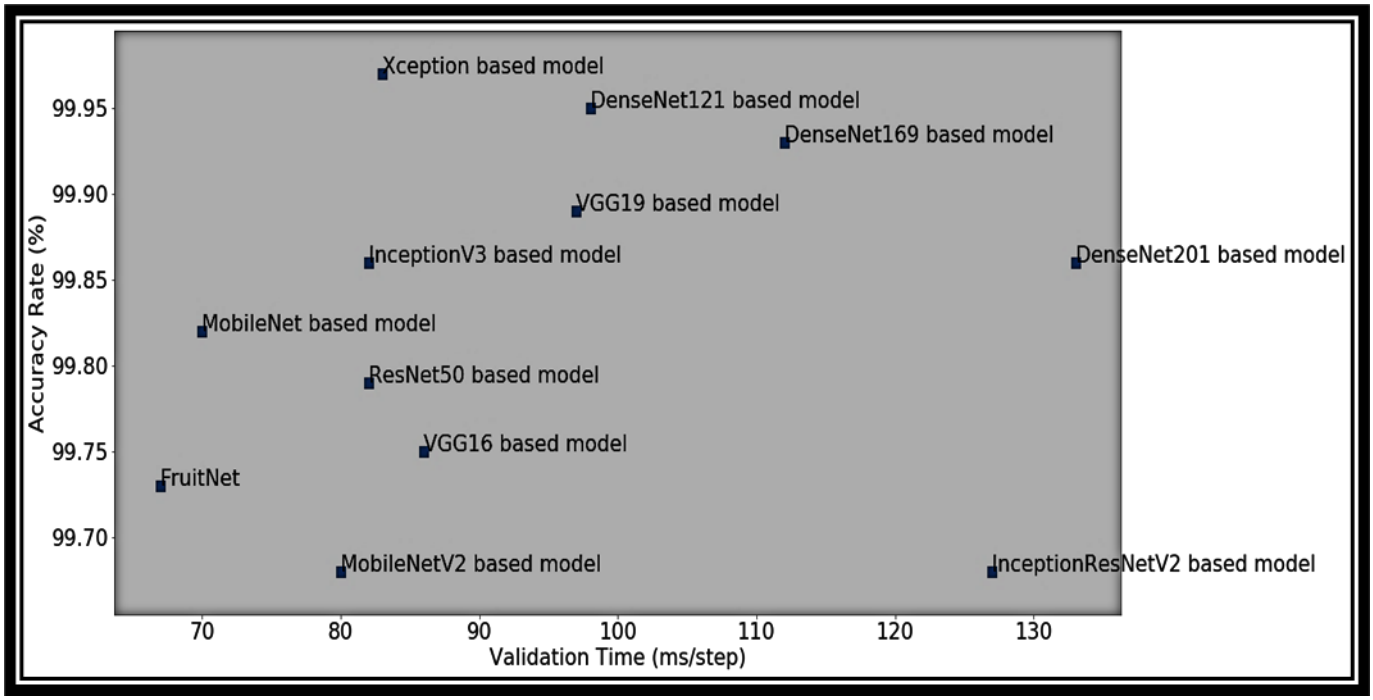

Figure 8. A scatter plot depicting accuracy/speed trade-off of the proposed deep learning models

accuracy. Such a high level accuracy rate has never been reported in the relevant literature [1, 2, 6, 8, 9, 10, 11, 12].

6 Conclusion

This paper proposes fourteen different deep learning based models for image based fruit type classification. Thirteen of these models exploit transfer learning or fine tuning i.e. feature extraction is done through some pre-trained feature extractor while final classification is performed by a custom classifier. In addition to these thirteen models, a simple, sequential 12-layer model called *FruitNet* has also been proposed. A large, publicly available fruit images dataset (called Fruits 360 [24]) has been used to carry out the experiments. The dataset contains 101 fruit classes. No past research work [1, 2, 6, 8, 9, 10, 11, 12, 33] was based on such a high number of fruit classes.

Among all the proposed models that exploited transfer learning, Xception based model has produced the best accuracy rate while MobileNet based model has proved to be a very efficient model for fruit image classification. Fine tuning (in the context of unfreezing some layers of the convolutional base) has been evaluated for fruit image classification and has been found to produce excellent results. The best classification accuracy of 99.98% is achieved when VGG16 based model is fine tuned. This accuracy rate is significantly higher than all the accuracy rates reported in the relevant literature [1, 2, 6, 8, 9, 10, 11, 12, 33]. Thus, the efficacy of fine-tuning in fruit image classification has been proved through these results. FruitNet, on the other hand, has given decent performance in terms of accuracy despite its very simple structure and small size. It has also proved to be the fastest model among all the proposed models. This is mainly due to its very lightweight feature extractor which consists of only four convolutional layers.

As for the future work, the author aims to evaluate the models on a more challenging dataset. Most fruit images of such a dataset should contain one or more of the following attributes:

1. Non-homogeneous, complicated background.
2. Partially occluded fruit(s)
3. Fruit(s) cut in half / sliced fruit(s)
4. Fruit(s) inside plastic bags
5. Unpicked fruit(s)
6. Fruit(s) with decay
7. Camera not well-focused on fruit(s), etc.

Apart from using a more challenging dataset, future research work should employ more advanced computer vision techniques / models (as and when they become available) so that fruit image classification systems can be developed for more and more realistic scenarios.

REFERENCES

[1] Anderson Rocha, Daniel C. Hauagge, Jacques Wainer and Siome Goldenstein. 2010. Automatic fruit and vegetable classification from images. *Computers and Electronics in Agriculture* 70, 1 (Jan. 2010), 96-104. DOI: https://doi.org/10.1016/j.compag.2009.09.002

[2] M. Shamim Hossain, Muneer Al-Hammadi and Ghulam Muhammad. 2019. Automatic Fruit Classification Using Deep Learning for Industrial Applications. *IEEE Transactions on Industrial Informatics* 15, 2 (Feb. 2019), 1027 - 1034. DOI: https://doi.org/10.1109/TII.2018.2875149

[3] Wenhao Shao, Yanjie Li, Songfeng Diao, Jingmin Jiang and Ruxiang Dong. 2017. Rapid classification of Chinese quince (Chaenomeles speciosa Nakai) fruit provenance by near-infrared spectroscopy and multivariate calibration. *Analytical and Bioanalytical Chemistry* 409, 1 (Jan. 2017), 115-120. DOI: https://doi.org/10.1007/s00216-016-9944-7

[4] Radi, S. Ciptohadijoyo, W. S. Litananda, M. Rivai and M. H. Purnomo. 2016. Electronic nose based on partition column integrated with gas sensor for fruit identification and classification. *Computers and Electronics in Agriculture* 121 (Feb. 2016), 429-435. DOI: https://doi.org/10.1016/j.compag.2015.11.013

[5] M. Fatih Adak and Nejat Yumusak. 2016. Classification of E-Nose Aroma Data of Four Fruit Types by ABC-Based Neural Network. *Sensors* 16, 3, Article 304 (Feb. 2016), 13 pages. DOI: https://doi.org/10.3390/s16030304

[6] Yu-Dong Zhang, Zhengchao Dong, Xianqing Chen, Wenjuan Jia, Sidan Du, Khan Muhammad and Shui-Hua Wang. 2019. Image based fruit category classification by 13-layer deep convolutional neural network and data augmentation. *Multimedia Tools and Applications* 78, 3 (Feb. 2019), 3613–3632, DOI: https://doi.org/10.1007/s11042-017-5243-3

[7] R. M. Bolle, J. H. Connell, N. Haas, R. Mohan and G. Taubin. 1996. VeggieVision: a produce recognition system. In *Proceedings of the 3rd IEEE Workshop on Applications of Computer Vision (WACV'96)*, December 2 - 4, 1996, Sarasota, FL, USA. IEEE. https://doi.org/10.1109/ACV.1996.572062

[8] Yudong Zhang and Lenan Wu. 2012. Classification of Fruits Using Computer Vision and a Multiclass Support Vector Machine. *Sensors* 12, 9 (Sep. 2012), 12489-12505. DOI: https://doi.org/10.3390/s120912489

[9] Yudong Zhang, Shuihua Wang, Genlin Ji and Preetha Philips. 2014. Fruit Classification using Computer Vision and Feedforward Neural Network. *Journal of Food Engineering* 143 (Dec. 2014), 167-177. DOI: https://doi.org/10.1016/j.jfoodeng.2014.07.001

[10] Shuihua Wang, Yudong Zhang, Genlin Ji, Jiquan Yang, Jianguo Wu and Ling Wei. 2015. Fruit Classification by Wavelet-Entropy and Feedforward Neural Network Trained by Fitness-Scaled Chaotic ABC and Biogeography-Based Optimization. *Entropy* 17, 8 (Aug. 2015), 5711-5728, DOI: https://doi.org/10.3390/e17085711

[11] Woo Chaw Seng and Seyed Hadi Mirisaee. 2009. A new method for fruits recognition system. In *2009 Int. Conf. on Electrical Engineering and Informatics*, August 5-7, 2009, Selangor, Malaysia. IEEE. DOI: https://doi.org/10.1109/ICEEI.2009.5254804

[12] Shui-Hua Wang and Yi Chen. 2018. Fruit category classification via an eight-layer convolutional neural network with parametric rectified linear unit and dropout technique. *Multimedia Tools and Applications*, (Sep. 2018), 1-17, DOI: https://doi.org/10.1007/s11042-018-6661-6

[13] Yann LeCun, Yoshua Bengio and Geoffrey Hinton. 2015. Deep learning. *Nature* 521, (May 2015), 436–444. DOI: https://doi.org/10.1038/nature14539

[14] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, June 20-25, 2009, Miami, FL. IEEE, 248-255. DOI: https://doi.org/10.1109/CVPR.2009.5206848

[15] François Chollet. 2017. Xception: Deep Learning With Depthwise Separable Convolutions. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 21-26, 2017, Honolulu, HI, USA. IEEE, 1251-1258. DOI: https://doi.org/10.1109/CVPR.2017.195

[16] Karen Simonyan and Andrew Zisserman. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv:1409.1556. Retrieved from https://arxiv.org/abs/1409.1556

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 27-30, 2016, Las Vegas, NV, USA. IEEE, 770-778. DOI: https://doi.org/10.1109/CVPR.2016.90

[18] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens and Zbigniew Wojna. 2016. Rethinking the Inception Architecture for Computer Vision. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 27-30, 2016, Las Vegas, NV, USA. IEEE, 2818-2826. DOI: https://doi.org/10.1109/CVPR.2016.308

[19] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke and Alex Alemi. 2016. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. arXiv:1602.07261. Retrieved from https://arxiv.org/abs/1602.07261

[20] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto and Hartwig Adam. 2017. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. arXiv:1704.04861. Retrieved from https://arxiv.org/abs/1704.04861

[21] Gao Huang, Zhuang Liu, Laurens van der Maaten and Kilian Q. Weinberger. 2018. Densely Connected Convolutional Networks. arXiv:1608.06993. Retrieved from https://arxiv.org/abs/1608.06993

[22] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov and Liang-Chieh Chen. 2019. MobileNetV2: Inverted Residuals and Linear Bottlenecks. arXiv:1801.04381. Retrieved from https://arxiv.org/abs/1801.04381

[23] Horea Muresan and Mihai Oltean. 2018. Fruit recognition from images using deep learning. *Acta Univ. Sapientiae, Informatica* 10, 1 (Aug. 2018), 26-42. DOI: https://doi.org/10.2478/ausi-2018-0002

[24] Fruits 360 Dataset on GitHub. Retrieved from https://github.com/Horea94/Fruit-Images-Dataset

[25] François Chollet. 2018. *Deep Learning with Python*. Manning Publications, New York, NY.

[26] Hoo-Chang Shin, Holger R. Roth, Mingchen Gao, Le Lu, Ziyue Xu, Isabella Nogues, Jianhua Yao, Daniel Mollura and Ronald M. Summers. 2016. Deep Convolutional Neural Networks for Computer-Aided Detection: CNN Architectures, Dataset Characteristics and Transfer Learning. *IEEE Transactions on Medical Imaging* 35, 5 (May 2016), 1285 - 1298. DOI: https://doi.org/10.1109/TMI.2016.2528162

[27] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke and Andrew Rabinovich. 2014. Going deeper with convolutions. arXiv:1409.4842. Retrieved from https://arxiv.org/abs/1409.4842

[28] Diederik P. Kingma and Jimmy Ba. 2017. Adam: A Method for Stochastic Optimization. arXiv:1412.6980. Retrieved from https://arxiv.org/abs/1412.6980

[29] Luis Perez and Jason Wang. 2017. The Effectiveness of Data Augmentation in Image Classification using Deep Learning. arXiv:1712.04621. Retrieved from https://arxiv.org/abs/1712.04621

[30] Alex Krizhevsky, Ilya Sutskever and Geoffrey E. Hinton. 2017. ImageNet classification with deep convolutional neural networks. *Communications of the ACM* 60, 6 (May 2017), 84-90. DOI: https://doi.org/10.1145/3065386

[31] Sebastien C. Wong, Adam Gatt, Victor Stamatescu and Mark D. McDonnell. 2016. Understanding Data Augmentation for Classification: When to Warp? arXiv:1609.08764. Retrieved from https://arxiv.org/abs/1609.08764

[32] François Chollet. 2019. Keras Documentation - Image Preprocessing. Retrieved from https://keras.io/preprocessing/image/

[33] Raheel Siddiqi. 2019. Effectiveness of Transfer Learning and Fine Tuning in Automated Fruit Image Classification. In *Proceedings of the 2019 3rd International Conference on Deep Learning Technologies*, July, 2019, Xiamen, China. ACM Inc., New York, NY, 91-100. DOI: https://doi.org/10.1145/3342999.3343002