**Fruit Type Detection and Classification using Advanced CNN**

1. **Exploratory Data Analysis (EDA): -** The data set used in this article is taken from ‘[Fruit Images for Object Detection](https://www.kaggle.com/mbkinaci/fruit-images-for-object-detection)’. This is data set consisting of 240 training images and 60 test images. Total number of images is 300 out of which we have 4 classes – Apple(94 images), Banana(91 images),Mixed(25 images) and Orange(90 images).

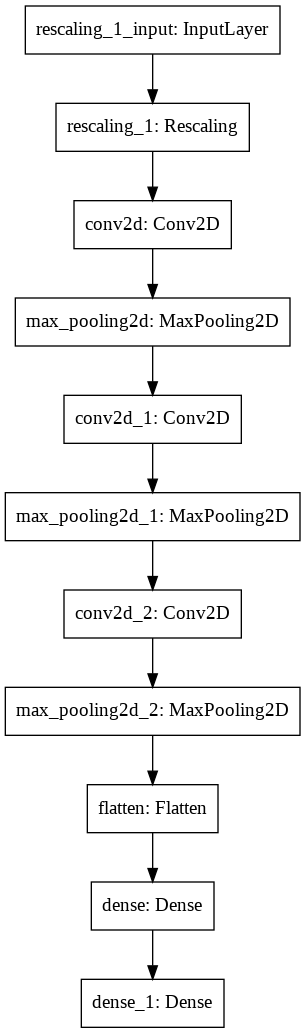
The images in dataset was not distributed evenly, hence made it in size of 180x180 pixels. RGB channel values are [0,255] range , but in neural network we can not use it, henc standardized the values to be in the [0, 1] range by using a Rescaling layer.

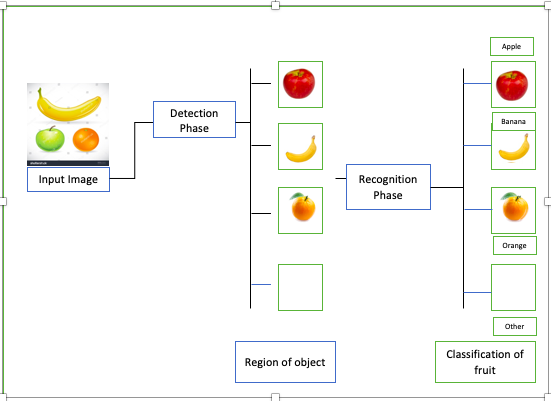
1. **Base Model & Architecture: -** In our model at first an input image is feed to the detection phase which outputs all the detected region of objects (ROO) of that image as separate images (k) each containing a single object and then every detected image (k) is feed to the recognition phase which distinguishes fruit ROO from non-fruit ROO and finally classify the fruit ROO to corresponding class of fruit. And non-fruit ROO as other object.

The detection process is all about identifying every single connected object from an image regardless of its location, quality of appearance, background of appearance and overlapping condition. Our proposed system can detect all connected objects from image. Here, a connected object refers to an object which contour (e.g., outline representing or bounding the shape of object) is a connected or closed entity. The recognition phase works for classifying every detected objects using CNN. Recognition phase mainly focuses on classifying each detected fruits from detection phase by their corresponding fruit class. In recognition phase nonfruit objects are marked as other object class. Therefore, they can be distinguished from fruits in an image.

The number of classes are 4. They are apple, orange, banana and other. We are using 3 Convolution layers, with maxpolling. The image values are scaled between 1 to 255 in the scaling layer of the model.

**Model Architecture:-**



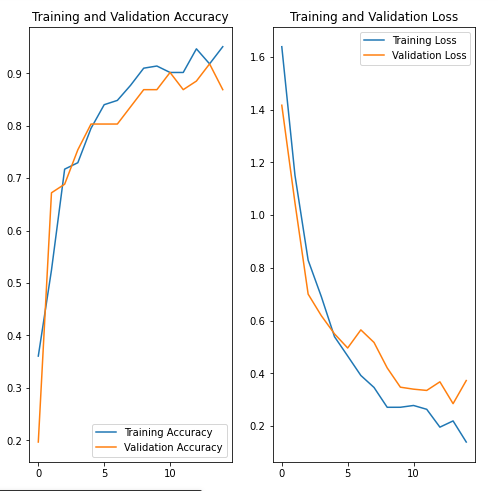
Th

**3. Early Results: -**

* There are 305 images belongs to 4 classes. The images are resized to (180, 180) before feeding into the model for training.
* The images are split into batches of size 32, with 20% as test images.
* Due to lack of data, the model is overfitting. We are getting train\_accuracy of 96% and test accuracy as 90%.
* Here are the results before data augmentation.



* The images are rotated, flipped, zoomed to create more data using builtin keras preprocessing functions. Also dropout layer is added to reduce the overfitting of the model.
* After Data Augmentation and adding Dropout layers, we are able to resolve the overfitting and able achieve train\_accuracy of 92% and test\_accuracy of 91% with 14epochs. Find the results below.



For the basic model we are using CNN, for better results we are planning to use R-CNN, Fast R-CNN, Yolo algorithms.

1. **Tentative list of Algorithms: -**

**R-CNN 2013: -** The Paper “Region-based Convolutional Networks for Accurate Object Detection and Segmentation” that talks about R-CNN is Rich feature hierarchies for accurate object detection and semantic segmentation which is popularly known as R-CNN. It was published in the year 2014. *Ross Girshick et al.*in 2013 proposed an architecture called R-CNN (Region-based CNN) to deal with this challenge of object detection. This R-CNN architecture uses the selective search algorithm that generates approximately *2000* region proposals. These *2000* region proposals are then provided to CNN architecture that computes CNN features. These features are then passed in an SVM model to classify the object present in the region proposal. An extra step is to perform a bounding box regressor to localize the objects present in the image more precisely.

**Fast R-CNN 2015:** - In R-CNN we passed each region proposal one by one in the CNN architecture and selective search generated around *2000* region proposal for an image. So, it is computationally expensive to train and even test the image using R-CNN. To deal with this problem Fast R-CNN was proposed, It takes the whole image and region proposals as input in its CNN architecture in one forward propagation. It also combines different parts of architecture (such as ConvNet, RoI pooling, and classification layer) in one complete architecture. (Paper: *Fast R-CNN*)

**Faster R-CNN 2015: -** Faster R-CNN was introduced in 2015 by *k He et al.* After the Fast R-CNN, the bottleneck of the architecture is selective search. Since it needs to generate *2000* proposals per image. It constitutes a major part of the training time of the whole architecture. In Faster R-CNN, it was replaced by the region proposal network. First of all, in this network, we passed the image into the backbone network. This backbone network generates a convolution feature map. These feature maps are then passed into the region proposal network. The region proposal network takes a feature map and generates the anchors (the center of the sliding window with a unique size and scale). These anchors are then passed into the classification layer (which classifies that there is an object or not) and the regression layer (which localize the bounding box associated with an object). (Paper: *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*)

**YOLO v1 2015:-** YOLO (You Only Look Once), is a network for object detection introduce in paper “[You Only Look Once: Unified, Real-Time Object Detection by Joseph Redmon, Santosh Divvala, Ross Girshick and Ali Farhadi (2015)](https://arxiv.org/abs/1506.02640)”. The object detection task consists in determining the location on the image where certain objects are present, as well as classifying those objects. Previous methods for this, like R-CNN and its variations, used a pipeline to perform this task in multiple steps. This can be slow to run and also hard to optimize, because each individual component must be trained separately. YOLO, does it all with a single neural network.

**SSD 2015:-** The paper about [SSD: Single Shot MultiBox Detector](https://arxiv.org/abs/1512.02325) (by C. Szegedy et al.) was released at the end of November 2016 and reached new records in terms of performance and precision for object detection tasks, scoring over 74% MAP (*mean* *Average Precision*) at 59 frames per second on standard datasets such as [PascalVOC](http://host.robots.ox.ac.uk/pascal/VOC/) and [COCO](http://cocodataset.org/#home).

**Single Shot:** this means that the tasks of object localization and classificationare done in a *single* *forward pass* of the network

**MultiBox:** this is the name of a technique for bounding box regression developed by Szegedy et al. (we will briefly cover it shortly)

**Detector:** The network is an object detector that also classifies those detected objects

**FPN 2016: -** paper, **FPN (Feature Pyramid Network)**, by **Facebook AI Research (FAIR)**, **Cornell University** and **Cornell Tech**, is reviewed. By introducing a clean and simple framework for building feature pyramids inside the convolutional neural network (CNN), significant improvements are shown over several strong baselines and competition winners such as [G-RMI](https://towardsdatascience.com/review-g-rmi-winner-in-2016-coco-detection-object-detection-af3f2eaf87e4), [MultiPathNet](https://towardsdatascience.com/review-multipath-mpn-1st-runner-up-in-2015-coco-detection-segmentation-object-detection-ea9741e7c413) and [ION](https://towardsdatascience.com/review-ion-inside-outside-net-2nd-runner-up-in-2015-coco-detection-object-detection-da19993f4766). And FPN has higher AR for segment proposals compared with [DeepMask](https://towardsdatascience.com/review-deepmask-instance-segmentation-30327a072339), [SharpMask](https://towardsdatascience.com/review-sharpmask-instance-segmentation-6509f7401a61) and [InstanceFCN](https://towardsdatascience.com/review-instancefcn-instance-sensitive-score-maps-instance-segmentation-dbfe67d4ee92).

**YOLO v2 2016:-** According to the paper of [YOLOv2](https://arxiv.org/pdf/1612.08242.pdf) “YOLO9000: Better, Faster, Stronger”, it became more accurate and faster than the previous version (YOLO). This is because YOLOv2 uses some techniques that YOLO didn’t use, such as **Batch-Normalization** and **Anchor-Boxes**. Batch-Normalization or BN is used to normalize the outputs of hidden layers. This makes learning much faster. Anchor-Boxes is assumption on the shapes of the bounding boxes. Since the shapes of objects, we’re trying to detect do not vary so much, we don’t have to find boxes that do not look like any of objects we want to detect.

**RetinaNet 2017: -** RetinaNet is a one-stage object detection model that utilizes a focal loss function to address class imbalance during training. Focal loss applies a modulating term to the cross-entropy loss in order to focus learning on hard negative examples. RetinaNet is a single, unified network composed of a *backbone* network and two task-specific *subnetworks*. The backbone is responsible for computing a convolutional feature map over an entire input image and is an off-the-self convolutional network. The first subnet performs convolutional object classification on the backbone's output; the second subnet performs convolutional bounding box regression. The two subnetworks feature a simple design that the authors propose specifically for one-stage, dense detection. (Paper: **Focal Loss for Dense Object Detection**)

**YOLO v3 2018:** - YOLO V3 is an improvement over previous YOLO detection networks. Compared to prior versions, it features multi-scale detection, stronger feature extractor network, and some changes in the loss function. As a result, this network can now detect many more targets from big too small. And, of course, just like other single-shot detectors, YOLO V3 also runs quite fast and makes real-time inference possible on GPU devices. (Paper: **YOLOv3: An Incremental Improvement**)

**Objects as Points 2019: -** Objects as Points, aka CenterNet, took a step further. It uses heat-map peaks to represent object centers, and the network will regress the box width and height directly from these box centers. Essentially, CenterNet is using every pixel as grid cells. With a Gaussian distributed heat-map, the training is also easier to converge compared with previous attempts which tried to regress bounding box size directly. (Paper : **Objects as Points**)

1. **References: -**

[1] Y. LeCUN, B. Boser, J. Denker, et al., “Hubbard. w., and jackel, ld:â˘AŸhand written digit recognition with a backpropagation network â˘AŸ, in â˘AŸ,” Advances in neural information processing systems 2, 396404 (1989).

[2] P. Sermanet, S. Chintala, and Y. LeCun, “Convolutional neural networks applied to house numbers digit classification,” in Pattern Recognition (ICPR), 2012 21st International Conference on, 3288–3291, IEEE (2012).

[3] D. Ciresan, U. Meier, and J. Schmidhuber, “Multicolumn deep neural networks for image classification,” (2012). [doi:10.1109/cvpr.2012.6248110].

[4] P. Sermanet and Y. LeCun, “Traffic sign recognition with multi-scale convolutional networks,” in Neural Networks (IJCNN), The 2011 International Joint Conference on, 2809–2813, IEEE (2011) [doi:10.1109/IJCNN.2011. 6033589].

[5] M. Everingham, L. Van Gool, C. K. Williams, et al., “The pascal visual object classes (voc) challenge,” International journal of computer vision 88(2), 303–338 (2010). [doi:10.1007/s11263-009-0275-4].

[6] J. Deng, W. Dong, R. Socher, et al., “Imagenet: A largescale hierarchical image database,” in Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, 248–255, Ieee (2009). [doi:10.1109/CVPR.2009.5206848].

[7] K. Jarrett, K. Kavukcuoglu, Y. LeCun, et al., “What is the best multi-stage architecture for object recognition?,” in Computer Vision, 2009 IEEE 12th International Conference on, 2146–2153, IEEE (2009). [doi:10.1109/ICCV.2009.5459469].

[8] B. V. Biradar and S. P. Shrikhande, “Flower detection and counting using morphological and segmentation technique,” Int. J. Comput. Sci. Inform. Technol 6, 2498– 2501 (2015).

[9] H. Mure¸san and M. Oltean, “Fruit recognition from images using deep learning,” Acta Universitatis Sapientiae, Informatica 10(1), 26–42 (2018).

[10] H. Kuang, C. Liu, L. L. H. Chan, et al., “Multi-class fruit detection based on image region selection and improved object proposals,” Neurocomputing 283, 241–255 (2018). [doi:10.1016/j.neucom.2017.12.057].

[11] Y. Lu, D. Allegra, M. Anthimopoulos, et al., “A multitask learning approach for meal assessment,” in Proceedings of the Joint Workshop on Multimedia for Cooking and Eating Activities and Multimedia Assisted Dietary Management, 46–52, ACM (2018).

[12] Kaggle, “Dogs vs. cats,create an algorithm to distinguish dogs from cats.” https://www.kaggle.com/c/dogs-vs-cats (2013).

[13] J. Rakun, D. Stajnko, and D. Zazula, “Detecting fruits in natural scenes by using spatial-frequency based texture analysis and multiview geometry,” Computers and Electronics in Agriculture 76(1), 80–88 (2011). [doi:10.1016/j.compag.2011.01.007].

[14] R. Thendral, A. Suhasini, and N. Senthil, “A comparative analysis of edge and color based segmentation for orange fruit recognition,” in Communications and Signal Processing (ICCSP), 2014 International Conference on, 463–466, IEEE (2014). [doi:10.1109/ICCSP.2014.6949884].

[15] E. Parrish, A. Goksel, et al., “Pictorial pattern recognition applied to fruit harvesting,” Transactions of the ASAE 20(5), 822–0827 (1977).

[16] G. R. A. Grand D Esnon and R.Pellenc, “A selfpropelled robot to pick apples,” in ASAE paper, (87-1037) (1987).

[17] S. Ren, K. He, R. Girshick, et al., “Faster r-cnn: Towards real-time object detection with region proposal networks,” in Advances in Neural Information Processing Systems 28, C. Cortes, N. D. Lawrence, D. D. Lee, et al., Eds., 91–99, Curran Associates, Inc. (2015).

[18] J. Redmon, S. Divvala, R. Girshick, et al., “You only look once: Unified, real-time object detection,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 779–788 (2016).

[19] R. Girshick, J. Donahue, T. Darrell, et al., “Rich feature hierarchies for accurate object detection and semantic segmentation,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 580–587 (2014).

[20] F. Garcia, J. Cervantes, A. Lopez, et al., “Fruit classification by extracting color chromaticity, shape and texture features: towards an application for supermarkets,” IEEE Latin America Transactions 14(7), 3434–3443 (2016). [doi:10.1109/TLA.2016.7587652].

[21] S. Riyadi, A. J. Ishak, M. M. Mustafa, et al., “Waveletbased feature extraction technique for fruit shape classification,” in Mechatronics and Its Applications, 2008. ISMA 2008. 5th International Symposium on, 1–5, IEEE (2008). "[doi:10.1109/ISMA.2008.4648858]".

[22] S. Jana and R. Parekh, “Shape-based fruit recognition and classification,” in International Conference on Multi Class Fruit Classification Using Efficient Object Detection and Recognition Techniques 17 Copyright © 2019 MECS I.J. Image, Graphics and Signal Processing, 2019, 8, 1-18 Computational Intelligence, Communications, and Business Analytics, 184–196, Springer (2017). [doi:10.1007/978-981-10-6430-2\_15].

[23] J.-y. Kim, M. Vogl, and S.-D. Kim, “A code based fruit recognition method via image convertion using multiple features,” in IT Convergence and Security (ICITCS), 2014 International Conference on, 1–4, IEEE (2014). [doi:10.1109/ICITCS.2014.7021706].

[24] C. Hung, J. Underwood, J. Nieto, et al., “A feature learning based approach for automated fruit yield estimation,” in Field and Service Robotics, 485–498, Springer (2015). [doi:10.1007/978-3-319-07488-7\_-33].

[25] Z. S. Pothen and S. Nuske, “Texture-based fruit detection via images using the smooth patterns on the fruit,” in Robotics and Automation (ICRA), 2016 IEEE International Conference on, 5171–5176, IEEE (2016). [doi:10.1109/ICRA.2016.7487722].

[26] H. N. Patel, R. Jain, and M. V. Joshi, “Fruit detection using improved multiple features based algorithm,” International journal of computer applications 13(2), 1–5 (2011).

[27] W. Liu, D. Anguelov, D. Erhan, et al., “Ssd: Single shot multibox detector,” in European conference on computer vision, 21–37, Springer (2016). [doi:10.1007/978-3-319- 46448-0\_2].

[28] H. Kuang, “Hulin kuang. the university of calgary..” https://www.researchgate.net/profile/Hulin\_Kuang.

[29] C. L. Zitnick and P. Dollár, “Edge boxes: Locating object proposals from edges,” in European conference on computer vision, 391–405, Springer (2014). [doi:10.1007/978-3-319-10602-1\_26].

[30] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, et al., “Object detection with discriminatively trained part-based models,” IEEE transactions on pattern analysis and machine intelligence 32(9), 1627–1645 (2010). [doi:10.1109/TPAMI.2009.167].

[31] Kaggle, “Fruits 360 dataset.” https://www.kaggle.com/moltean/fruits.

[32] L. Hou, Q. Wu, Q. Sun, et al., “Fruit recognition based on convolution neural network,” in Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD), 2016 12th International Conference on, 18–22, IEEE (2016). [doi:10.1109/FSKD.2016.7603144].

[33] L. Jidong, Z. De-An, J. Wei, et al., “Recognition of apple fruit in natural environment,” Optik-International Journal for Light and Electron Optics 127(3), 1354–1362 (2016). [doi:10.1016/j.ijleo.2015.10.177].

[34] R. Khan, T. Fariha Raisa, and R. Debnath, “An efficient contour based fine-grained algorithm for multi category object detection,” Journal of Image and Graphics 6, 127– 136 (2018).

[35] E. S. Gastal and M. M. Oliveira, “Domain transform for edge-aware image and video processing,” in ACM Transactions on Graphics (ToG), 30(4), 69, ACM (2011). [doi:10.1145/2010324.1964964].

[36] E. Gastal, “Non photorealistic rendering using opencv( python, c++ ) | learn opencv.” https://www.learnopencv.com/nonphotorealisticrendering- using-opencv-python-c/.

[37] P. O. A. Thresholding, “Adaptive Thresholdings,” (2003). [Online; last accessed 06-April-2019].

[38] S. F. BogoToBogo\_K Hong Ph.D.Golden Gate Ave, “Image thresholding and segmentation..” https://www.bogotobogo.com/python/OpenCV\_Python/py thon\_opencv3\_Image\_Global\_Thresholding\_Adaptive\_T hresholding\_Otsus\_Binarization\_Segmentations.php (2013). [Online;accessed 19-July-2018].

[39] Homepages.inf.ed.ac.uk, “Morphology - Dilation,” (2003). [Online; last accessed 06-April-2019].

[40] O. S. C. V. OpenCV, “Morphological transformations.” https://docs.opencv.org/3.4/d9/d61/tutorial\_py\_morpholog ical\_ops.html.