**FLOWER CLASSIFICATION USING CONVOLUTIONAL NEURAL NETWORK**

**Joanna Marie Diaz**

***Quality Assurance Engineer***

***joannamarie838@gmail.com***

**Abstract:** Image classification has become an essential study in Machine Learning. From health industry as to identify types and classes of diseases to self-driving cars. It is a technological evolution that creates advancement and reliability on different industry. Flower classification is one of the challenging tasks in the fields of computer vision and taxonomy. It involves consideration of some features like the flowers flexible form, background illumination and the similarities from its classes. Due to this, I conclude that selecting to create a flower classification model can be beneficial for my occurring studies of machine learning.

For this task I used the same 20 species of flowers base from the Oxford-102 flower datasets. The datasets were composed of 40 – 250 images so I added some images to the species I selected to add for model complexity. I used TensorFlow’s Keras library to build a flower classification model to train and test the model based on the images from the web. The model has 4 convolution layers and 2 fully connected layers. By steadily specifying and tuning the model hyperparameters I get the fair result of the train and test set by 82% for both accuracy and val\_accuracy eluding overfitting and under fitting of the data to the model.

**1. Introduction**

Image classification is considered as one of the challenging supervised learning problems. It is implemented by training a machine learning model that can distinguish each class of object from an image by analyzing the image data correlation. The development of a model to imitate how a human perceives an object and make classification and prediction, is regarded as deep learning. There are some challenges that affect the training of an image classification model like intra-class variation, scale variation, view-point variation, occlusion, illumination, and background clutter (B. R. Mete and T. Ensari, 2019). The implementation of Convolutional Neural Network reduces some obstacles of classifying the object in an image. It has made an image classification model produce a more accurate result.

One of the challenging tasks for computer vision is developing a model that classify the diversity of flower species. Flowers have a wide range of species, that

concludes similar color, shape, appearance and surrounding objects like leaves and grass (Hiary et al, 2018).

Flowering plants are considered one of the most diverse species in botany. The number of species of flowering plants is approximated to be in the range of 250,000 to 400,000. Most plants have their beneficial substance, if one can understand what kind of plants are sprouting in their backyard or in the environmental places. Many have limited knowledge when it comes to recognizing flower species, mostly the rare and obscure ones. Even the experts in the field must use a specific key (Dichotomous and synoptic key) data to identify which group the flower belongs to. And they regard a picture inadequate to identify its variety. Hence, it is even more challenging for non-experts to identify a specific flower more so for a machine. But it was proven by a lot of experts that identifying an object in a picture can be obtained by a machine learning model by the means of various techniques and algorithms. As stated by B. R. Mete and T. Ensari, (2019), designing a machine learning model that will recognize rare plant species will be beneficial in fields such as pharmaceutical industry, botany, and trade activities. Being able to identify a plant or flower species just by taking a picture of a particular flower species using your smart phone could be an amenity for people who evaluate it for natural advancement on the field and even more by someone who is mesmerize to the beauty of nature, such contrivances is of a great aid.

**2. Related Work**

I was first inspired on the study that is made by the British statistician and biologist Ronald Fisher where he collected the data to evaluate the morphologic variation of Iris flowers of three related species. The data comparison shows tremendous correlation of differences between the three species using supervised learning and unsupervised learning classification technique. Though the study did not use images to determine the classification for the Iris flower species, it could still be considered as a good comprehension on importance of determining a plant species.

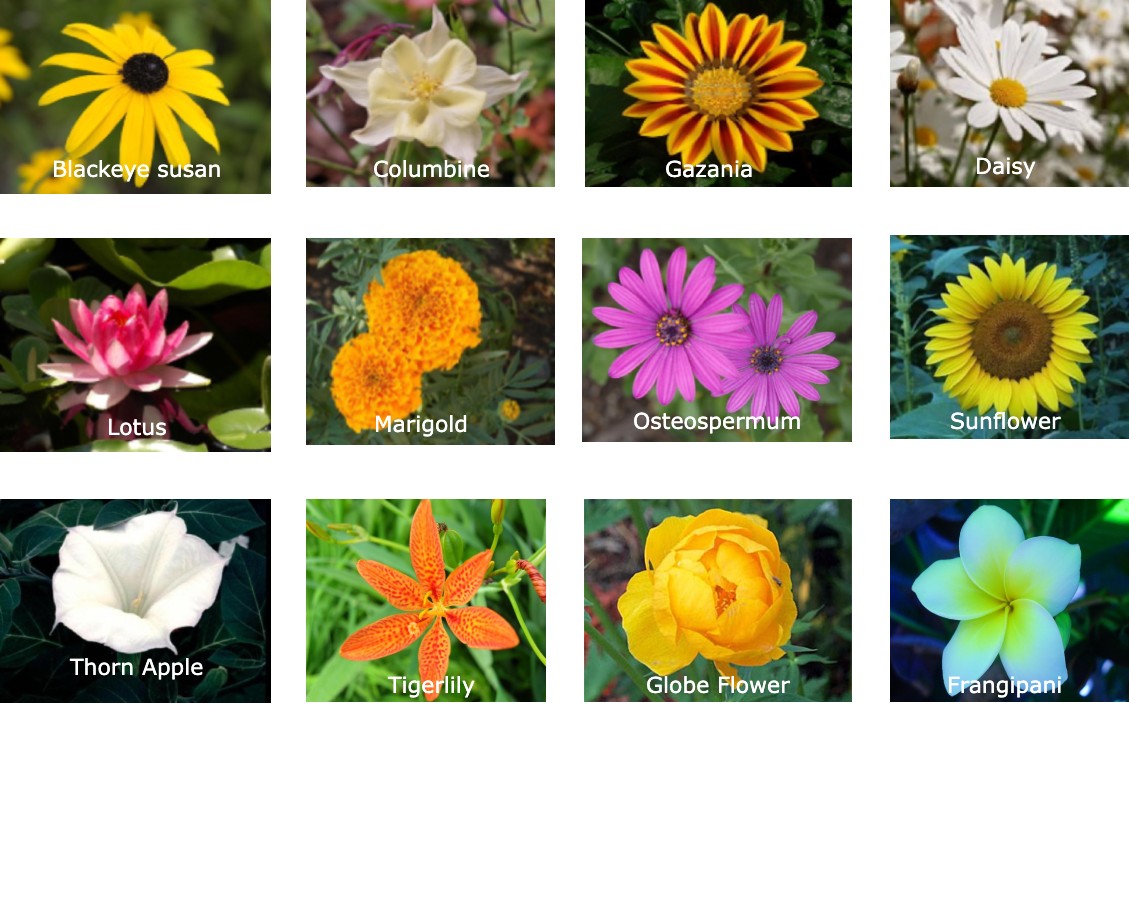
Many research and studies were conducted for flower classification using different techniques for image classification and the most effective one is using deep neural network to train a classification model. One of the notable study for flower classification are the research conducted by Oxford Visual Geometry Group called 102-Flowers (that is based on their previous study called 17-Flowers) where they distinguish four different properties of flowers using multiple kernel framework with SVM classifier and color features (SIFT, HOG, HSV) where each describe various aspects namely, the local shape/texture, the shape of the boundary, the overall spatial distribution of petals and the color (Nilsback, M-E. and Zisserman, A., 2008). They use multiple kernel framework with Support Vector Machine as classifier. The dataset is composed of 102 distinct species of flowers and 8189 images. The method achieves an accuracy rate of 72.8%.

The same dataset was also used by various researchers like Liu et al (2016) who proposed a deep learning CNN algorithm for flower classification for the same dataset. The proposed architecture has 5 convolutional layer and 3 fully connected layer and achieve an accuracy rate of 84%.

Studies using transfer learning were also conducted for flower classification. One is where they use the train CNN model and use various transfer learning architectures like VGG16, ResNet50 and MobileNet (Narvekar and Rao, 2020). The accuracy rate for all the trained models (CNN, VGG16, ResNet50) is on an average of 93% aside from the MobileNet with an average accuracy rate of 82%. Xia et al (2017), increased the accuracy rate from the initial machine learning model using transfer learning of GoogleNet inception-v3 architecture and get an accuracy rate of 95% and 94% accordingly. While Patel and Patel (2020) use different techniques of transfer learning method and Deep Convolutional Neural Network, NAS-FPN and Faster R-CNN, to classify 30 flower species datasets and 102-flower dataset that sum up to 19679 of flower images. The results are 87.6% for 102-flower dataset and 96.2% for 30 species datasets. Hence, I also based the CNN model on the 102-flower datasets.

**3. Dataset**

I use the Oxford-102 Flower dataset as my reference for the 20 species of flower for the training of CNN model then segregate and label them according to the name of their species.

**Sample Flower Species in the dataset

**4. Building the model**

Convolutional Neural Network is a structured recognition technique which has been developed in recent years. This network averts the intricate preprocessing of the image that results for an actual image to be processed directly. It uses local receptive field, weights sharing and pooling technology that makes the training parameters reduce compared to the neural network. It also has a certain degree of translation, rotation, and distortion invariance of image. It has made immense progress in the field of computer vision (Xia et al, 2017).

Convolutional Neural Network has a different architecture from a regular neural network. The neurons in CNN are arranged in 3 dimensions which are the width, height, and depth. For this study, I resize the images to a tensor size of 150x150x 3 to achieve neutral value for each image.

For Convolutional Neural Network image classification, setting hidden layer is necessary due that the data are obviously non-linearly separated. But determining how many hidden layers that a model should process and setting the value for hyperparameters is a task that can only be obtained with rigorous evaluation.

**4.1.Convolution Layer**

The main building block of a ConvNet is the convolutional layer. It is also called feature extraction stage due that it processes two functions by means of a kernel or filter to produce a new function called feature map. The filter slides through all the dataset until it computes the last data from the previous layer. It is a linear computation that involves the computation of the dot product between a dimensional array of input data and a set of dimensional arrays of weights that produce a scalar output of an activation map (Albawi et al., 2017).

The computational expression for the kernel can be expressed on this equation:



Formula for Convolution Layer. [Source](https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939)

Where:

W – input size (height or width size)

F – spatial size

P – padding

S – number of strides

For this phase I initially set the filters to 32, 64, 96 and 128 respectively. With kernel size of 3x3, ‘same’ padding and Rectified Linear Unit (ReLU) as activation function.

**4.2. Pooling Layer**

Pooling layer reduces the dimensionality of the feature map to reduce computing time and tackle overfitting. On the process of dimensional reduction, the neurons are being subjected to invariance translation or producing an output layer that emphasizes the important portion of the input layer with minor changes to the principal value (Brownlee, 2019).

There are several pooling functions but the most eminent one is the max pooling function. Which acquires the maximum value from the previous layer which can be occupied by the window on a specified rate of each stride. Max pooling is commonly used mostly on a CNN model. The size of pooling operation is smaller than the feature map, frequently the size is set to

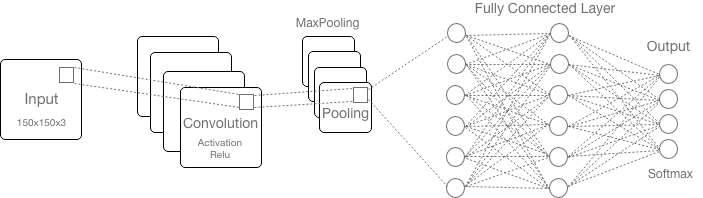
2x2 with a stride of 2 pixels, extracting the feature map to one quarter a size.

**4.3. Fully connected layer**

Fully connected layer has two main phases. First is the step where it obtains a flattened 1-dimensional data from the preceding layer while performing vector matrix multiplication on each neuron. Second is where neurons are calculated and produced an activation value for the final classifier that is Softmax. Softmax function takes an N dimensional value and converts it to another N dimensional value ranging from 0 to 1. It generates a computed probability for the output layer.

**4.4. Model Architecture**

By implementing a multi-class classifier, I first try to reduce the complexity of the model and set the convolution and pooling layer to 4 layers and add 3 fully connected layers. The activation function for the hidden layer is set to ReLu and softmax for the output layer.



Proposed CNN model architecture

**5. Data Preparation and Preprocessing**

Declaring the label for each flower category is crucial for training a CNN model. It is the first step that I must consider extracting the possible outcome once the model has been trained and tested. I segregate each flower species on different directory rather than naming the file one by one. Subsequently use the label as a predictor of the model.

Feature scaling is a necessary step to take prior to training the model to ensure that all features in the data have the same scale (Wan, 2019). For this the image array pixel has been normalized using OpenCV normalized function and concurrently perform one hot label encoding using scikit-learn LabelEncoder for the label.

**5.1. Optimization**

To be able to get a proper prediction, the model must learn the right amount of weight and bias that will be distributed in the network. In this process the difference between the predicted output and the expected output is minimized to get an acceptable probability of prediction (Goodfellow et al., 2017). Gradient descent is the algorithm used to adjust the weight and bias of the model based on the hyperparameters that is provided.

Here I initially try to use mini-batch gradient descent for the model. Where the batch size is greater than one sample of data and less than training set.

**5.2. Hyperparameters Tuning**

Hyperparameter tuning is defined as finding a set of optimal value to produce a close to accurate prediction of a model (Mishra, 2020). The hyperparameters are set to feed for the chosen algorithm on different stages of the model. Setting the hyperparameters manually takes considerable time and computer resources nevertheless using Gridsearch is not feasible for training a Neural Network as it has large datasets and each training requires plenty of time to acquire the result.

The following are the value for each task during Hyperparameters tuning. The first 5 tuning are test with 70% training set while the last 5 is set to 75% training set.

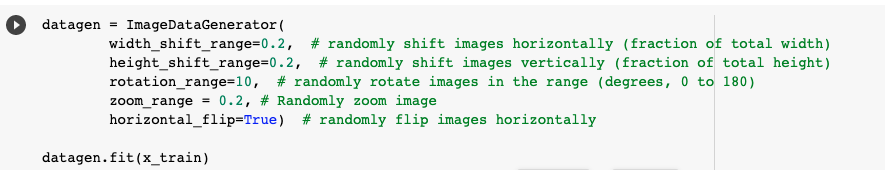
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tasks** | **Epochs** | **Batch size** | **Learning Rate** | **Drop out (2 FC)** |
| 1 | 100 | 64 | 0.001 | [0.5, 0.8] |
| 2 | 80 | 50 | 0.002 | [0.5, 0.8] |
| 3 | 100 | 64 | 0.002 | [0.7, 0.8] |
| 4 | 80 | 70 | 0.001 | [0.7, 0.8] |
| 5 | 100 | 70 | 0.001 | [0.6, 0.8] |
| 6 | 100 | 60 | 0.002 | [0.6, 0.8] |
| 7 | 100 | 64 | 0.002 | [0.7, 0.8] |
| 8 | 100 | 60 | 0.001 | [0.8, 0.8] |
| 9 | 120 | 50 | 0.001 | [0.7, 0.9] |
| 10 | 120 | 60 | 0.001 | [0.7, 0.9] |

**5.3. Data Augmentation**

Data augmentation can help increase the size of the dataset and prevent overfitting. The datasets that are provided have less amount of data than the model has a possibility of encountering overfitting. Images are high dimensional and include enormous variety of factors of variation, data augmentation can also help the model generalize the image better by looking at various aspects of the image (Kusnetzky, 2011).

Keras library has provided better access for augmenting data. ImageDataGenerator class has attributes to produce different orientation of input images.

The following are used to augment each image: random rotation of 0-to-180-degree angle, enlarge image by 25%, create shifted images by 10% vertically and horizontally and finally flipping the image horizontally. Here is the sample image orientation.

Code snippet:

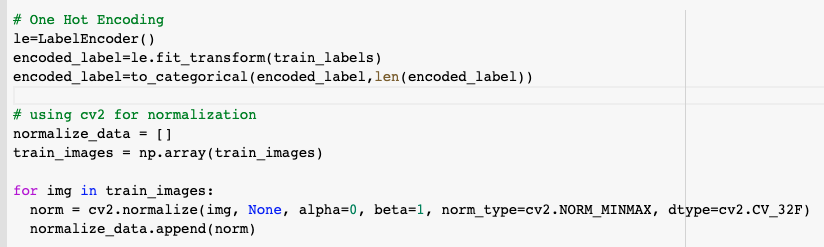
Sample Augmented Image

**5.4. One Hot Encoding and Normalization**

“There is no exclusive accurate value for the depth of an architecture, just as there is no exclusive accurate value for the length of a computer program. The ideal network architecture for a task must be found via experimentation guided by monitoring the validation set error.” (Goodfellow et al, 2017).

For the model to effectively classify each image, specifying a label for each image category is imperative. In this step, I created different directories on each flower’s species name. After creating label and segregating each image, the label will be used for training and perform one hot encoding using sklearn’s LabelEncoder to transform categorical label into integer value.

Data normalization is a necessary phase which guarantee that each input parameter (pixel, in this event) has a similar data distribution and centralized evident image data. This makes concurrence faster while training the network. It is performed by subtracting the mean from each pixel and then dividing the outcome by the standard deviation (Nikhil, 2017). This also make image more appealing by adding contrast to images.

This process is implemented using OpenCV’s normalize function.

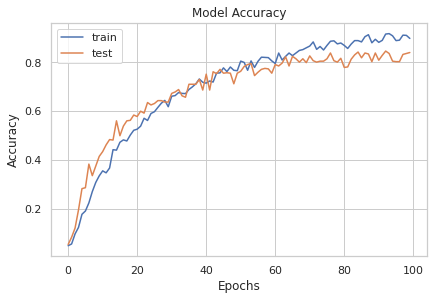
**6. Training and Testing**

After preparing the data and declaring the feature and label variable, the data is prepared for training and testing using scikit-learn train\_test\_split. The training size is 70% and 75% while 20% and 25% for testing. Since the random state is set to 42, the seed value had to also be set to 42 to resolve the randomness of the result.

**7. Visualization**

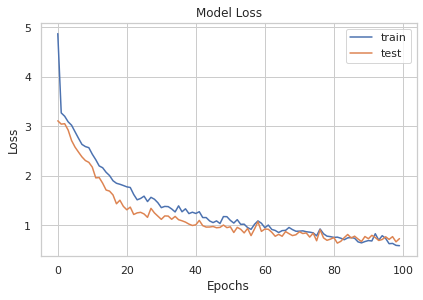
While tuning the hyperparameters of the model, over fitting and under fitting has encountered.

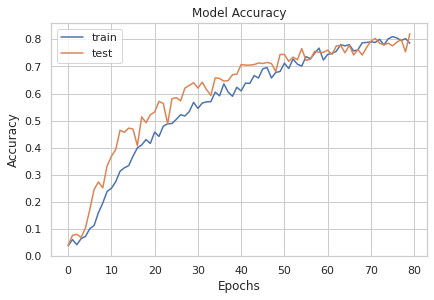
**7.1. Overfitting Result**

****

****

**7.2. Under fitting Result**



****

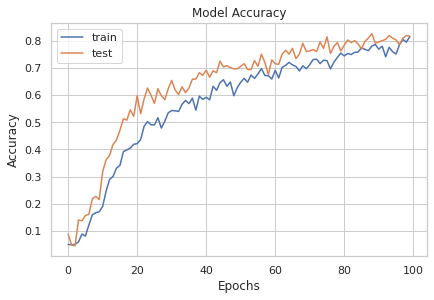
**8. Result**

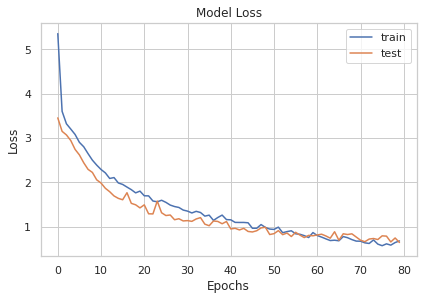
Having a complex architecture is not suited with small datasets. It either adds more images or reduces the complexity of the model. Due to this I adjust some hyperparameters on convolution like filters and kernel size and remove one fully connected layer. I also adjusted the batch size from the default value of 50 to 70 and epoch size from 80 to 120.

By discretely adjusting some hyperparameter, I get an accuracy for train set from 74% to 90%, the test accuracy has dropped from 76% to 85% and loss from 0.86 to 0.28 while val\_loss from 0.87 to 0.64. Between this value an overfitting and under fitting was encountered.

The best by far hyperparameter value that I found that is fitted with the model are the following:

* Epochs: 100
* Batch size: 50
* Learning Rate: 0.001
* Dropout: 0.7 for first FC and 0.8 for second FC

The result was fairly 83% for both accuracy and val\_accuracy with a loss average value of 0.68.

****

**8.1. Prediction**

During model prediction, the test set were predicted based on the trained model. The counted result of correctly classified images is 359 while 64 images were mis-classified.

**9. Conclusion**

Image classification is one of the challenging tasks for machine learning. Mostly when there is a lot of interference for the image classification. The concept of Convolutional Neural Network made the task less intricate but there is still a lot of propound refinement to get a better prediction with less human intervention.

The trained model in this study still needs adequate value for prediction improvement that could produce a justifiable result.

For this study I conclude that being able to create a model that can classify an image is a substantial knowledge not only in the field of computer technology but also in various industrial innovation.

**10. Implemented Code Repository**

<https://github.com/j-anne/ML_Final_Assessment.git>

**11. References**

Albawi, S., Mohammed, T.A. and Al-Zawi, S., 2017, August. Understanding of a convolutional neural network. In *2017 international conference on engineering and technology (ICET)* (pp. 1-6). Ieee.

Almogdady, H., Manaseer, S. and Hiary, H., 2018. A flower recognition system based on image processing and neural networks. *International Journal Of Scientific & Technology Research*, *7*(11), pp.166-173.

Bengio, Y., 2012. Practical recommendations for gradient-based training of deep architectures. In *Neural networks: Tricks of the trade* (pp. 437-478). Springer, Berlin, Heidelberg.

Brownlee J., 2019, Machine Learning Mastery, Available at: <https://machinelearningmastery.com/> (Accessed: 22 July 2022)

B. R. Mete and T. Ensari, 2019, "Flower Classification with Deep CNN and Machine Learning Algorithms," 2019 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), 2019, pp. 1-5, doi: 10.1109/ISMSIT.2019.8932908.

Goodfellow, I., Bengio, Y. and Courville, A., 2017. Deep learning (adaptive computation and machine learning series). *Cambridge Massachusetts*.

Gulli, A. and Pal, S., 2017. Deep learning with Keras. Packt Publishing Ltd.

Hiary, H., Saadeh, H., Saadeh, M. and Yaqub, M., 2018. Flower classification using deep convolutional neural networks. *IET Computer Vision*, *12*(6), pp.855-862.

Liu, Y., Tang, F., Zhou, D., Meng, Y. and Dong, W., 2016, November. Flower classification via convolutional neural network*.* In 2016 IEEE International Conference on Functional-Structural Plant Growth Modeling, Simulation, Visualization and Applications (FSPMA) (pp. 110-116). IEEE.

Mishra M., 2020, Convolutioal Neural Networks Explained, Available at: <https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939>, (Accessed: 26 July 2022)

Narvekar, C. and Rao, M., 2020, December. Flower classification using CNN and transfer learning in CNN-Agriculture Perspective. In *2020 3rd International Conference on Intelligent Sustainable Systems (ICISS)* (pp. 660-664). IEEE.

Nikhil B., 2017, Image Data pre-Processing for Neural Networks, available at: <https://becominghuman.ai/image-data-pre-processing-for-neural-networks-498289068258> (Accessed: 1 Feb 2023)

Nilsback, M-E. and Zisserman, A., 2008, Automated flower classification over a large number of classes, Visual Geometry Group, Department of Engineering Science University of Oxford, United Kingdom

Patel, I. and Patel, S., 2020. An Optimized Deep Learning Model For Flower Classification Using NAS-FPN And Faster R-CNN. *International Journal of Scientific & Technology Research, 9(03)*, pp.5308-5318.

Plantlet, 2022, Key: A Plant Identifying Method, available at: <https://plantlet.org/plant-identification-terms/> (Accessed: 20 July 2022)

Standford University, CS231n Convolutional Neural Network for Visual Recognition*,* Available at: <https://cs231n.github.io/convolutional-networks/> (Accessed: 23 July 2022)

Xia, X., Xu, C. and Nan, B., 2017, June. Inception-v3 for flower classification. In *2017 2nd international conference on image, vision and computing (ICIVC)* (pp. 783-787). IEEE.

Wan, X., 2019, June. Influence of feature scaling on convergence of gradient iterative algorithm. In Journal of physics: Conference series (Vol. 1213, No. 3, p. 032021). IOP Publishing.