A FLOWER RECOGNITION SYSTEM USING DEEP NEURAL NETWORK COUPLED WITH VISUAL GEOMETRY GROUP 19 ARCHITECTURE

Zi Yuan Ong\textsuperscript{1*}, Kah Kien Chye\textsuperscript{1}, Huay Wen Kang\textsuperscript{1} and Chi Wee Tan\textsuperscript{1}

\textsuperscript{1} Faculty of Computing And Information Technology, Tunku Abdul Rahman University College, Kampus Utama, Jalan Genting Kelang, 53300, Wilayah Persekutuan Kuala Lumpur, Malaysia
*Corresponding author: ongzy-wp17@student.tarc.edu.my

ABSTRACT

Computer vision is one of the basic features to streamline processes like robotic process automation and digital asset management. Computer vision has come a long way in terms of its capabilities and what it can provide and do for different industries. Object detection and image detection are just some of the few applications provided by computer vision. However, this field is still relatively young and prone to challenges. The first is the lack of well-annotated images to train the algorithms to perform optimally, and the second being lack of accuracy when applied to real-world images different from the ones from the training dataset. As such, this paper aims to fine-tune pre-trained machine learning models, which are ResNet50 and VGG19 as well as training a new SqueezeNet inspired model from scratch to create a flower recognition model that can process and remember large amounts of flower species data. In conclusion, VGG19 was found to perform the best on both the 5 Categories and Flower-102 dataset, with an accuracy of 88% and 84% respectively.

Keywords: VGG19, Transfer Learning, Deep Learning, Flower Recognition, Neural Network

1.0 INTRODUCTION

There are approximately 369,000 named flowering plant species in the world (Liu et al., 2016). In general, experienced plant taxonomists can identify plants based on the features of flowers such as sepals, petals, stamens, and carpels. However, most people find it tough to determine these flowers apart. Additionally, someone may be confused with similar flower species. This is where object recognition comes in, as it is able to understand and analyze images effortlessly and instantaneously. Therefore, the main objective of this project is to develop a flower recognition model that can correctly identify the class of flowers. The flower recognition model will analyse the image and to identify whether an input image contains a certain type of flower. The goal is to train a computer to do what comes naturally to humans, which is to comprehend what is included in an image and provide insight from it. This project is interested in fine-tuning pre-trained machine learning models as well as training a flower recognition model from scratch. It’s essential to not only aim for novel and innovative methods but also conduct in-depth research on existing methods. This could lead to better insights and new discoveries.
2.0 LITERATURE REVIEW

Lv et al. (2021) has proposed a flower classification model based on saliency detection and optimised VGG-16 deep neural network model tested on Oxford Flower-102 data set. To improve the model, optimization algorithm of stochastic gradient descent was done which can also reduce resource consumption and training time. Dropout method was used to reduce model overfit by randomly discarding training information. The use of transfer learning techniques was also done to solve the problem of insufficient image data which can also reduce model training time. This model shows an accuracy of 91.9%, which is higher than other traditional methods for image classification tasks and proves the feasibility of flower identification.

Cibuk et al. (2019) employed pre-trained DCNN models for feature extraction. They chose two popular DCNN models, AlexNet and VGG-16, and concatenated features from both models to construct efficient feature sets. They used the minimum Redundancy Maximum Relevance (mRMR) model to act as a feature selection algorithm. Then, a support vector machine (SVM) classifier with Radial Bases Function (RBF) kernel is employed to classify the flower species using the extracted features. Their experimental results showed that they managed to achieve a 96.39% and 95.70% accuracy performance for the Flower-17 and Flower-102 dataset respectively.

Gogul and Kumar (2017) has proposed a flower classification approach based on the Inception-v3 model of the Tensorflow platform, using transfer learning technology to retrain the flower categories. They have tested this approach using three models, the Inception-v3 model, the Xception model and the OverFeat model. On top of that, they have used a machine learning classifier such as Logistic Regression or Random Forest on top of the CNN models to increase the accuracy rate. Their approach minimizes the hardware requirements needed to perform the computationally intensive task of training a CNN. This approach outperforms all the handcrafted feature extraction methods such as Local Binary Pattern (LBP), Color Channel Statistics, Color Histograms, Haralick Texture, Hu Moments and Zernike Moments. This paper yields impressive Rank-1 accuracies of 73.05%, 93.41% and 90.60% using OverFeat, Inception-v3 and Xception architectures, respectively as Feature Extractors on Flower-102 dataset.

Liu et al. (2016) proposed a flower classification approach using a convolutional neural network to extract features. They have also obtained the luminance map which is generated by converting RGB pixels to YUV, and the brightness of the color is extracted from the Y component, which allows better performance as flowers have high brightness. They use a regional contrast based salient object detection algorithm to compute a bottom-up saliency map, which simultaneously evaluates global contrast difference and spatial weighted coherence scores. The algorithm is simple, efficient, naturally multi-scale, and produces full-resolution, high-quality saliency maps which enhances the performance. They have achieved 76.54% accuracy in their dataset, and 84.02% in the Oxford Flower-102 dataset.

2.1 Existing Method

SqueezeNet was selected as the model to be used for this flower recognition project. SqueezeNet is an innovative convolutional neural network which has 112 times fewer parameters than another CNN, Alexnet, while also maintaining an accuracy top-5 performance comparable to that of AlexNet. Being a small model, SqueezeNet is more amenable to on-chip implementations on FPGAs (Iandola et al., 2016). Research has been done using the SqueezeNet model in various use cases, and the results were promising. Sayed,
Soliman and Hassanien (2021) implemented a model to predict melanoma skin cancer evaluated on ISIC 2020 and ISIC 2019, and uses a SqueezeNet model optimised with a bald eagle search (BES) optimization to find the best hyperparameter. The proposed melanoma skin cancer prediction model obtained an overall accuracy of 98.37%, specificity of 96.47%, sensitivity of 100%, f-score of 98.40%, and area under the curve of 99%. The experimental results showed the robustness and efficiency of the proposed model compared with VGG-19, GoogleNet, and ResNet50.

ResNet, short for Residual Networks, allows engineers to train hundreds or even thousands of layers while still achieving impressive results. In 2015, this model won the ImageNet challenge. It has been studied that the increasing training error of deep neural networks is due to the network's initialization, optimization function, or one of the most well-known problems, the vanishing gradient problem (He et al., 2016). It is a problem that happens during the training of artificial neural networks using gradient-based learning and backpropagation. Gradients are known and used to update the weights in a network during backpropagation. However, sometimes the gradient becomes vanishingly small, effectively preventing the weights from changing values. This causes the network to stop training because the same values are propagated over and over again, resulting in no useful work being done. Residual neural networks are used to solve such problems. ResNet employs skip connections to add the output of an earlier layer to a later layer, thereby mitigating the vanishing gradient problem (He et al., 2016).

VGG is an innovative object-recognition model which supports up to 19 layers. It is pre-trained with ImageNet datasets and is still able to outperform with other unseen datasets which make it one of the most used image recognition architectures. There are multiple variants for the VGGNet including VGG-16 and VGG-19, where these variants only differ in the total number of layers in the neural network. Multiple research has been done by using the VGG-19 model and impressive results were obtained. Victor Ikechukwu et al. (2021) has performed experiments using ResNet-50, ChexNet, VGG-19 and their own proposed Iyke-Net to identify pneumonia from chest x-ray images, where VGG-19 achieved a high accuracy of 93.5%, coming in close second after their proposed Iyke-Net which is 93.6% accurate.

3.0 METHODOLOGY
This section talks about the datasets used and the design of the models.

3.1 Datasets
The first dataset is the Kaggle flower recognition dataset consists of 4242 images from flickr, google images and yandex images (Mamaev, 2021). The images are divided into five classes: daisy, tulip, rose, sunflower and dandelion. There are about 800 images for each class, each image about 320x240 pixels. The photos are not reduced to a single size but come with different proportions.

The Oxford Flower-102 dataset consisting of 102 more specific flower categories is also used (Nilsback & Zisserman, 2008). The images are flowers that are commonly found in the United Kingdom. There are between 40 to 258 images for each class, where each image has various scales, pose and light. There are large variations within the same category and several very similar categories which increase the difficulty of classification. The dataset has a total of 8189 images.
3.2 Model Architecture

The first model is inspired by the original SqueezeNet model. In basic, it is a partial implementation of the original SqueezeNet (Iandola et al., 2016). It makes use of the Fire modules architecture as designed by the original authors. A fire module consists of a squeeze convolution layer of only 1x1 filters, which feeds into an expand layer that has a mix of 1x1 and 3x3 convolution filters. The liberal use of 1x1 filters greatly reduces the parameters as a 1x1 filter has 9 times fewer parameters than a 3x3 filter. The squeeze layer decreases the number of input channels to 3x3 filters, also greatly reducing the parameters in the layer. Thus, SqueezeNet is able to maintain reasonable accuracy while being more than 50 times smaller than another model, AlexNet and exceeding AlexNet’s top-1 and top-5 accuracy on the ImageNet dataset. This particular implementation is a stripped down version of SqueezeNet with fewer layers. It consists of one input into a conv2d layer, followed by batch normalization, the first fire module, the first MaxPooling2D layer, the second fire module, the second MaxPooling2D layer, the third fire module, the first GlobalAveragePooling2D layer and the final Dense layer with softmax activation to obtain the categories to be predicted.

ResNet50 is chosen for transfer learning because of its lower computational power and promising accuracy compared to different ResNet variants like ResNet18, ResNet34, ResNet101 and ResNet152 (He et al., 2016). Firstly, the pre-trained ResNet50 model (resnet50 weights tf dim ordering tf kernels.h5) is downloaded from Github (Fchollet, 2016) and uses the weights from this downloaded model which trained from the imagenet datasets. Then, the first layer of ResNet50 is frozen and makes it non-trainable. If the first layer is trainable, the model may take a long time during the training process because it will have more trainable parameters. In this project, the pre-trained model weights should not be retrained because they are the advantage while taking the transfer learnt model. Besides, when a pre-trained network is used for transfer learning, additional dense layers shall be added at the end of the pre-trained network in order to learn which combination of the previously learned features helps in recognising the objects in the new dataset. Thus, additional layers were implemented after the output of the ResNet50 model. A total number of 7 layers were added in this project, that is the flatten layer, batch normalization layers, two customised layers with size 2048 and 1024, accompanied with “Rectified linear unit” (ReLU) activation function and the softmax layer. The flatten layer converts data into a 1-dimensional array for input to the next layer, and batch normalisation is a layer that allows each layer of the network to learn more independently. Finally, a softmax layer is included as the output layer in order to predict fixed types of flowers whereby only 5 or 102 classes of classification are produced, depending on the dataset. The 5 classes are produced to predict the 5 types of flowers for the 5 category dataset, which are daisy, tulip, rose, sunflower and dandelion. So, after applying this layer, a transfer learning model is created that can classify the input images into various types of flowers based on predictions from the pre-trained ResNet50 model.

VGG-19 is composed of 16 convolutional layers, 5 pooling layers, 3 fully-connected layers and a final layer of softmax function (Simonyan & Zisserman, 2015). The matrix was shaped (224, 224, 3) as the fixed input size of (224x224) RGB image is passed into this network. Kernel size of (3x3) instead of large kernels with stride of 1 pixel is used to cover every part of the image, whereas a 2x2 pixel window with a stride of 2 pixels is used to perform max pooling. The multiple layers of small kernels are able to effectively cover the
images without the use of large kernels such as 11x11 kernel in AlexNet and 7x7 kernel in ZFNet. Therefore, the number of parameters is also reduced and the overfitting problem is reduced. All hidden layers are equipped with ReLU to introduce non-linearity for better classification compared to previous models which used tanh or sigmoid functions. In this project, the pretrained weights are used by setting the parameter weights to imagenet. The default classifier is removed by setting `include_top` to false so a new classifier can be created. The first 19 layers are frozen to prevent the weights from being modified. Similar to the ResNet50 model, additional layers are added after the pretrained model. A max-pool layer is added to down-sample the input features. A flattening layer is then added before a dense layer with a softmax function since the dense layer accepts 2D input.

### 3.3 Logical Flow

![Logical Flowchart](image)

**Figure 1.** Logical Flowchart.

**Data Loading:** The flowers dataset consists of images of flowers with different class labels and stored in respective directories. This stage is to load the data from their directories and concatenate it into one dataframe. As a result, the original flower images end up with an image dimension of 244x244, which minimises image dimensions while maintaining image readability with efficient computational complexity and accommodating the input shape for the pre-trained models. For the 5 category dataset, there are 4242 images and 5 class labels. For the 102 category dataset, there are 8189 images and 102 class labels.
**Data Understanding:** The flowers dataset contains examples of labelled flower images. Each example includes a JPEG flower image as well as the class label. The exploration in the image data helps to validate the class distribution of each type of flower and ensures a balanced dataset, preventing an imbalanced dataset that leads to poor prediction. Besides, data visualization is required to sample and examine the input data to ensure image readability by randomly previewing the 10 images in the dataframe via 2D representation with the new image dimensions.

**Data Labelling:** This stage is used to transform categorical data (textual data) into numerical values for the prediction functions so that the deep learning predictive models can understand. Label encoding technique is performed to convert categorical values to numbers in this step.

**Model Creation:** The same optimizer and loss function is used to compile all models. Cross-entropy is the loss function chosen to evaluate a set of weights in multi-class classification problems for flower recognition. Furthermore, the Adam optimizer with the default learning rate (0.01) is used to search through different weights for the network. Finally, because this is a classification problem, classification accuracy is collected and reported, which will be defined via the metrics argument.

**Data Shuffling:** This stage is used to redistribute the training and testing data samples in the dataset to ensure that each data sample produces an "independent" change on the model, without being influenced by the points that came before it. Since the images were added sequentially from subfolders into the dataframe during data loading, data shuffling is required. Otherwise, the model can only learn what is "daisy" from the first 800 images, which does not optimise the model's parameters. The seed is set to 100 and is applied to both images and labels to ensure that each image matches the correct label.

**Data Augmentation:** This stage is used to increase the amount of data by adding slightly modified copies of already existing data or newly created synthetic data from existing data. This refers to randomly changing the images in ways that shouldn’t impact their interpretation, such as horizontal flipping, zooming, and rotating. Through data augmentation, it acts as a regularizer and prevents overfitting when training a machine learning model. It is the technique used to overcome the problem of overfitting by creating more data and making the model generalize well on the unseen data.

**Model Training:** The model is using the predefined train-test split for the Oxford Flower-102 dataset, with 1020 training samples and 6149 testing samples. The 5 category dataset is split into train-test sets using a ratio of 80:20. Different batch sizes and epochs are adjusted in different models in order to achieve the optimal result. This stage generates tensor image data in batches, which will be looped over for both training and testing. The neural network in each model takes in inputs, which are then processed in hidden layers using weights that are adjusted during training. Then the model spits out a prediction. The weights are adjusted to find patterns in order to make better predictions.

**Model Evaluation:** In this stage, a learning curve and confusion matrix is presented to evaluate the model performance. There are four aspects to be compared: training accuracy, training loss, validation accuracy and validation loss in every epoch. Usually, with every epoch increasing, the loss should be going lower and accuracy should be going higher. The validation loss and validation accuracy measures were calculated after the model had gone through all the data. So the network had been fully trained when these scores were calculated. On the other hand, the confusion matrix is a summary of classification problem prediction
results. The number of correct and incorrect predictions for each flower class is summarised with count values and broken down by each class.

4.0 RESULTS AND DISCUSSION

This section shows the results obtained for each model, on both datasets. The metric used is the validation accuracy as metrics like training accuracy do not reflect real-life performance. The validation accuracy is calculated by taking the True Prediction/Total Number of Predictions using the validation dataset. The models were validated with \( N = 1020 \) samples for the Oxford Flower-102 dataset and \( N= 848 \) samples for the 5 category dataset.

<table>
<thead>
<tr>
<th>Table 1. Validation accuracy for each model for the two datasets.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Validation Accuracy (5 Category)</td>
</tr>
<tr>
<td>----------------------------------</td>
</tr>
<tr>
<td>SqueezeNet Inspired Model</td>
</tr>
<tr>
<td>ResNet50</td>
</tr>
<tr>
<td>VGG19</td>
</tr>
</tbody>
</table>

Based on our results, the partially pre-trained VGG-19 model performed the best on both datasets, achieving a high validation accuracy of 88% and 84% on the 5 Categories and Flower-102 dataset respectively. This may be the result of the simple structures of VGG-19 and its hidden layers with ReLU function that can better introduce non-linearity for better classification compared to other models. The smaller number of features produced also contributes to the better generalization of the model.

The SqueezeNet Inspired model had the second highest validation accuracy of 77% and 67% on the 5 Categories and Flower-102 dataset respectively. This might be due to the fact that while this SqueezeNet Inspired model is far less complex and has less performance than both other models, this model is not pretrained. Instead, the whole model was trained solely on the two datasets individually for each scenario, and therefore was able to perform better than the partially pre-trained ResNet50. This is a lightweight model, even when compared to the original SqueezeNet implementation, and therefore it should be expected that there were sacrifices in terms of performance.

Finally, the ResNet50 model has the least performance, achieving 67% and 42% on the 5 Categories and Flower-102 dataset respectively. This could be due to the ResNet50 model's architecture being overly complicated for this task, leading to poor generalisation. The selection of optimal learning rate, batch size and identifying the best freezing layer is playing an important role for the better performance of the model.

5.0 CONCLUSION

The objectives are fulfilled as a model using VGG19 has been created that can perform flower classification tasks with 89% accuracy with 5 categories of flowers, and 84% accuracy on 102 species of flowers. However, there are limitations in this research such as limitations in hardware. As many of the algorithms and models in this project are all computationally intensive. This project is also more concerned with which models perform better on the two datasets of flowers, and not with exploring why the models perform the way that they do.
Another potential avenue for exploration is the other pretrained models like AlexNet, VGG16 and various other models, which are not used in this project. More comprehensive research comparing even more models can be done in the future to ensure completeness.

6.0 ACKNOWLEDGEMENTS
Authors thank the Faculty of Computing and Information Technology, Tunku Abdul Rahman University College for financial support and resources to carry out this project.

REFERENCES
Iandola, F. , Han, S., Moskewicz, M., Ashraf, K., Dally, W. and Keutzer, K. (2016) ‘SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size’.