## Data Preprocessing

### Data Overview

There are 4xxx images among 8 classes in our dataset. We split the dataset into two subsets: the training set accounts for 80% of the dataset, and the remaining is the validation set. These numbers follow the best practices in training machine learning model [].

### EDA

### Data Generation

Figure 1 indicates there are two problems in our dataset. First, xxx is the class having the highest number of samples (xxx), while xxx has the least figure (xxx). Therefore, our dataset is not general enough because according to this resource, 1000 images per class is suggested for training a deep model. Secondly, our dataset is imbalanced because there is a huge difference between these two above classes. We need to address this problem because the low generalization of the dataset can cause underperformance of our model due to biases. To overcome this situation, our target is to generate about 1500 images for each type of flower. First, we use the Selenium package to scrap the data from popular flower shops. This is the most effective way to gain more images, enriching the generalization of the trained model. We estimate that we collected more than 200 images in each class; these numbers are not enough to fulfill our dataset. Hence, we apply image augmentation on all existing images. Next, we augment these images by using the Albumentations package. This method increases the number of samples and diversify the labeled training sets by transforming the input image into vary forms []. After applying image augmentation, we have 11xxx images in our dataset. Figure 2 illustrates the distribution of image in each class after generation, and the number of images in each class ranges from xxx to xxx. Moreover, another advantage of image augmentation is that we create some images under a special condition, such as sun flare or low light. Therefore, adding these images can improve our models’ generalization because these models were trained under a more complicated data. Improving generalization is important because it helps model improve its performance in real life.

## Task 1: Classify Flower Images

### Evaluation Framework

There are three reasons why we choose accuracy as our main metric to evaluate model performance. The first reason is that the problem of imbalanced dataset has been resolved in the previous section. Hence, the situation of a model achieving a high accuracy by overfitting a specific class is unlikely. Moreover, we do not consider the importance of a particular kind of flower. Therefore, the sake of recall or precision is not as important as accuracy. If we are approaching a medical based problem, where the crucial of a specific class is highly recognized, and the dataset is imbalanced, recall or precision must have a higher priority than accuracy. Finally, we found that accuracy is the main metric to evaluate the Flower-102 dataset, which is an official dataset for evaluating a novel architecture.

### Strategy

We follow Karpathy’s recipe for training a model: baseline models, overfit, and regularize. Therefore, we overfit a particular model on the dataset until it achieves the train accuracy of more than 90%. For implementing this method, the model must learn on an uncomplicated dataset. Hence, we train the model on the generated dataset without applying any augmentations. This way helps the model gain a high accuracy faster compared to applying any image augmentation methods. Moreover, the model has learned some characteristics of the dataset before. Therefore, it can converge more quickly during learning some harder datasets (by applying image augmentation). This idea is like that of domain adaptation because the model has gained some knowledge about data distributions. After that, we gradually increase the strength of the augmentation. This helps the model performance gain the complex features of the dataset optimally and avoid the situation of overfitting during training.

### Configurations

#### Batch Normalization (BN)

Internal covariate shift is a phenomenon in which the distribution of input of each layer changes during training a deep Neural Network [1]. This will slow down and make it harder to train models. One way to resolve this is to normalize layers input and to say in other words, we normalize a part of the structure of the models and it is called Batch Normalization [2]. It acts as a regularizer and helps models to achieve the same accuracy with fewer training steps, hence, saving time as well [3].

#### Dropout

In the process of training a Neural Network, when the data is limited, our model can be overfitted, which results in bad performances. There have been many methods to reduce or to prevent overfitting in the model such as weight penalties L1 and L2 regularization and soft weight sharing [4]. However, to save time finding optimal hyperparameters for each different structure to regularize, dropout technique is introduced to address this problem. This technique includes dropping out a unit, temporarily removing it out of the network, along with its connections as well, refer to Fig. 1. However, there are drawbacks when applying dropout. It will increase your training time so in other words, applying dropout also means creating a trade-off between accuracy and training time.

#### AdamW and SGD

AdamW stands for Adam with decoupled weight decay. Both L2 regularization and weight decay regularization aims to reduce overfitting in models but how they work are slightly different. In a conference paper at ICLR 2019 by Loshchilov & Hutter, Adam with L2 regularization had been proved to have worse generalization performance than AdamW [6]. The paper also mentioned that AdamW had yielded better training loss and test error than Adam [7].

#### GeLU

GELUs stands for Gaussian Error Linear Units and it is a high-performing neural

network activation function [8]. As the Neural Network goes deeper and deeper while training, sigmoid activation has been proved to be less effective than RELUs (Rectified Linear Units), one of the most popular Machine Learning activation [9]. Based on the success of RELUs, ELUs was introduced as an activation to increase training speed. Based on the paper of Hendrycks & Gimpel in 2020, GELUs had been proven to yield higher accuracy than RELUs and ELUs.

#### He Initialization

He initialization, also known as Kaiming initialization is an initialization introduce in 2015 [10]. This method will allow us to train deep rectified models from scratch and look into deeper architectures [11]

#### Cross Entropy Loss

### Architectures

#### AlexNet

AlexNet is a groundbreaking convolutional neural network architecture that changed the field of deep learning. Developed by Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton in 2012, AlexNet revolutionized the field of computer vision by outperforming all the other models in the 2012 ImageNet Large-Scale Visual Recognition Challenge. AlexNet is also notable for introducing the use of ReLU (rectified linear unit) activation functions, which help to improve the speed of the network.

The AlexNet architecture is built with eight weighted layers: five convolutional layers and three fully-connected layers. The convolutional layers are responsible for extracting features from the input image, while the sub-sampling layers are used for down-sampling it. ReLU activation functions are used for all the convolutional layers as well as the fully-connected layer.

There are various advantages to the AlexNet architecture. First, with AlexNet, images are directly input to the classification model. AlexNet also makes use of numerous prominent industry approaches, such as the dropout, GPU, parallel processing, and ReLU. Furthermore, convolution layers can extract picture edges automatically, and fully-connected layers can learn these features.

Despite its advantages, the AlexNet architecture has certain drawbacks. In comparison to newer models such as ResNet, GoogLeNet, and VGGNet, the architecture is not as deep. Following that, the use of big convolution filters (5\*5) has been discouraged. Furthermore, using the normal distribution to start the weights in neural networks cannot successfully tackle the problem of gradient vanishing, which is later replaced by the Xavier technique.

#### VGG

VGG (Visual Geometry Group) is a multi-layered deep convolutional neural network that was introduced in 2015 by K. Simonyan and A. Zisserman as a method to boost network depth through the use of small (3\*3) convolution kernels. VGG has advantages over AlexNet since its small kernel size reduces the number of parameters in the model significantly. This reduces the model's complexity and allows the model to learn faster. Moreover, VGG utilizes the power of (1\*1) convolution layers to enhance the non-linearity features of the input data. This allows models to learn more complex features on the existing data and make more reliable decisions.

The VGG-11 is VGG model that is built with eleven weighted layers: eight convolutional layers and three fully connected layers. Additionally, the VGG network contains a total of five pooling layers scattered under distinct convolutional layers.

There are various advantages to VGG architecture. First, it is relatively small and efficient. VGG-11 is a deep network with only 11 layers, which makes it significantly more efficient than alternative architectures like ResNet or Inception. This makes it suitable for applications where computational resources are limited. Second, VGG-11 is extremely potent. Compared to GoogleNet, in terms of single-net basis, the VGG16 architecture performs 0.9 percent better (7.0% test error).

Despite its advantages, VGG architecture has certain drawbacks. First, the training process takes a significant amount of time. Second, VGG model is computationally expensive due to its depth and number of fully-connected nodes. This can be a concern in applications where computational resources are limited.

#### ResNet

ResNet (Residual Neural Network), developed by Microsoft Research in 2015, is a groundbreaking artificial neural network (ANN) that builds a network by stacking residual blocks on top of each other. There are several varieties of ResNet, all with different quantity of layers, but which all rely on the same principle. ResNet50 is a design variant that simplifies convolutional neural networks while addressing the problem of diminishing accuracy by using shortcut connections.

As the name suggested, ResNet50 is a variation that makes use of 50 layers: 48 Convolution layers, a Max Pool layer, and an Average Pool layer.

There are various advantages to the ResNet architecture. First, with ResNet, training networks with numerous layers (up to thousands) is simple and doesn’t increase the training error percentage. Second, by utilizing identity mapping, ResNet50 contributes in the resolution of the vanishing gradient problem.

Despite its advantages, the ResNet architecture has certain drawbacks. One issue is that it requires a lot of data to be effective. This means that it is not suitable for smaller data sets. Additionally, ResNet is prone to overfitting, which can lead to decreased accuracy in the model.

#### MobileNet

MobileNet is a deep learning-base architecture developed by Google based on a streamlined architecture that uses depth-wise separable convolutions to build light weight deep neural networks. It was created to help make deep learning more accessible to mobile and embedded vision applications.

The MobileNet architecture is based on depth wise separable convolutions. This type of convolution replaces the traditional convolution with two independent operations. The first operation is a depthwise convolution in which an identical filter is applied to each input channel. The second operation is a pointwise convolution, which combines the output of the depthwise convolution with a 1x1 convolution. This allows MobileNet to reduce the number of parameters while maintaining the same accuracy.

There are various advantages to the MobileNet architecture. First, MobileNet is a cost-effective solution to stay connected and access the internet while on the go. It is widely available, and customers can benefit from service packages that are tailored to their specific requirements. Second, MobileNet is more dependable and provides a better connection than other mobile networks, and users can be confident that their connection will not be dropped in the middle of important tasks.

Despite its advantages, the MobileNet architecture has certain drawbacks. The biggest disadvantage of MobileNet is that it is not as fast as other networks. This can be a problem for those who need a faster connection, and those who need to access large files quickly. Also, MobileNet's coverage is not as extensive as that of other networks, which can be troublesome for individuals who are constantly on the move.

#### EfficientNetV2

EfficientNetV2 is an advanced convolutional neural network (CNN) architecture that builds upon the success of its predecessor, EfficientNet. One notable improvement is the introduction of a new compound scaling approach known as "EfficientNetV2 Compound Scaling." This method scales the network's depth, width, and resolution simultaneously, enabling it to achieve optimal performance across different computational resources and constraints.

The EfficientNetV2 architecture is divided into multiple stages, each containing a series of convolutional layers, non-linear activation functions, and down-sampling operations. These stages are organized in a hierarchical manner, with lower stages capturing low-level visual features and higher stages capturing more abstract and complex features. This hierarchical design allows EfficientNetV2 to extract meaningful representations from input images at multiple scales, increasing its discriminative power.

There are various advantages to the EfficientNetV2architecture. First, EfficientNetV2 follows an efficient design philosophy, with the goal of achieving excellent performance with fewer parameters and computational resources. Furthermore, the Residual Channel Attention Network (ReCAN) integration in EfficientNetV2 model's ability to capture fine-grained details and boosts accuracy. ReCAN recalibrates feature maps using channel-wise attention, allowing the model to focus on informative regions while suppressing noise or irrelevant features.

Despite its advantages, the EfficientNetV2 architecture has certain drawbacks. First, training and inference periods are longer due to the increased depth and width of the network. Second, EfficientNetV2's complicated architecture can make it difficult to interpret and comprehend the model's inner workings. Furthermore, the performance of EfficientNetV2 is based on the quality and diversity of the training data. In other words, it may not perform as well on datasets that differ greatly from the ones on which it was trained highlighting the importance of data representation and transfer learning.

#### DarkNet53

DarkNet53 is a convolutional neural network architecture that was introduced as the foundation of the YOLOv3 object detection model in 2016. DarkNet53 is also an enhancement of DarkNet19, which debuted in the YOLOv2 model.

The DarkNet53 architecture is named after its network depth, consisting of 53 convolutional layers. It follows a simple and efficient design philosophy, focusing on stacking multiple layers with small filter sizes to capture increasingly complex features. This design choice enables DarkNet53 to extract high-level representations while minimizing computational complexity.

There are various advantages to the DarkNet53 architecture. First, the inclusion of residual connections in DarkNet53 alleviates the vanishing gradient problem and facilitates the training of deeper networks. This allows the architecture to effectively learn and tune its parameters, leading to enhanced performance. Moreover, DarkNet53 employs feature fusion from numerous scales, allowing the model to catch objects of varying sizes. The architecture preserves both fine-grained details and global context by merging features from several levels, boosting its ability to detect objects reliably.

Despite its advantages, the DarkNet53 architecture has certain drawbacks. While DarkNet53 incorporates skip connections to capture contextual information, it may not capture long-range dependencies as effectively as some other architectures.

#### CSPDarkNet53

CSPDarkNet53 is an extension of the DarkNet architecture that implements Cross Stage Partial Network (CSPNet) across different stages of the network. These connections allow the flow of information between early and late stages, facilitating better information propagation and feature reuse.

The CSPDarkNet53 architecture employs a CSPNet strategy to partition the feature map of the base layer into two parts and then merges them through a cross-stage hierarchy.

There are various advantages to the CSPDarkNet53 architecture. CSPDarkNet53 provides a training time advantage, allowing for faster convergence and reduced training durations compared to other architectures.

Despite its advantages, the CSPDarkNet53 architecture has certain drawbacks. It is worth noting that CSPDarkNet53 has a trade-off in terms of accuracy, as it may not match the levels achieved by certain alternative architectures, thereby presenting a limitation in its applicability.

### Ultimate Judgement

### Evaluation

## Task 2: Recommend Flower Images

### Evaluation Framework

It is prevalent for using precision and recall rates to measure the performance of a Content-based Image Retrieval (CBIR) system. Precision (specificity) measures the capability of the model to retrieve images like the given input image. Regarding recall, it assesses the ability to retrieve related images compared to the whole related images in the saved database. The reason why we do not consider accuracy is because accuracy makes the CBIR system become a binary classifier with two classes (relevant and non-relevant). Moreover, accuracy can lead to misunderstanding by recognizing all items as non-relevant because the number of non-relevant samples is significantly larger than the figure of relevant ones. Furthermore, this is not an effective way in image retrieval since the user always wants to acquire the relevant information.

There are some popular metrics for image comparison, such as L1, L2, Cosine, and Coefficient. The problem of using L1 and L2 for measuring image similarity is that it cannot handle the problem of brightness variance in the images. Moving onto Cosine, although this metric can solve the problem of bright contrast in image comparison, but it is sensitive when there are biases in the brightness of such images Regarding Cosine and Coefficient, although Cosine Similarity can solve the problem of bright contrast, we choose Coefficient as the main metric because this metric can handle the problem of bias effectively. This advantage of Coefficient makes it the best metric for solving the problem of brightness variance in assessing image similarity.

### Architecture

The issue of these above metrics is they cannot effectively indicate the images’ features. This is because these metrics evaluate the image similarity based on pixel-wise-pixel. Moreover, these pixels are nearly independent of each other, which makes it hard for such metrics to indicate the characteristics of each image. Another problem of applying those metrics naively is high computational resources. The reason is these metrics evaluate all the pixel of the image, leading to wasting computational resources. Hence, we resolve the problem of extreme calculation by implementing an encoder to reduce the image dimensionality. The encoder reduces the dimension of each image. Thus, our metrics can perform their calculation on a smaller amount of data, declining the resources for the evaluation process. Additionally, the information after reducing is more correlated with each other. This means that this information captures the characteristics of each image sharply. At this moment, our metrics can evaluate the similarity more precisely.

A picture containing text, screenshot, font, rectangle

Description automatically generated

Figure 1 CBIR System Architecture

Figure X shows our CBIR architecture. To begin with, we apply an extractor on our image database. At this stage, the extractor extracts the feature’s characteristics, indexes all of images, and stores the extracted information into a local file. The reason we precaculate the image’s features is we want to significantly avoid the number of redundant calculation resources for each image query. Next, for each time retrieving images, the extractor gains the input image’s characteristics before calculating the similarity score between each image. Next, we apply the K-nearest neighbors algorithm to find the 10-most similar images, rank all of them, before returning the results. Below we represent two ways we implement the extractor.

#### Extractor as Convolutional Layers in CNN Classifier

We implement these layers in task 1 for reducing the image dimensionality to indicate the features of each image. This is because these networks have learnt the dataset’s features by performing the classification task. Therefore, they have some understanding about the distribution of our dataset and make suggestions based on these insights. The advantage of this method is that we can utilize the models in task 1 for completing the mission of task 2.

#### Extractor as Encoder in Autoencoder

High-dimensional data can be converted to low-dimensional codes by training in a multilayer network with a small central layer to reconstruct high-dimensional input vectors. Gradient descent can be used for fine-tuning the weights in such “autoencoder” networks, but this works well only if the initial weights are close to good solution. We describe an effective way of initializing the weights that allows deep autoencoder networks to learn low-dimensional codes that work much better than principal components analysis as a tool to reduce dimensionality of data.

Dimensionality reduction facilities the classification, visualization, communication, and storage of high-dimensional data. A simple and widely used method is PCA, which finds the directions of greatest variance in the dataset and represents each data points by its corrdinates along each of these directions. We describe a nonlinear generalization of PCA that uses an adaptive, multilayer encoder network to transform the high-dimensional data into low-dimensional code and similar “decoder” network to recover data form the code.

Starting with random weights in the two networks, they can be trained together by minimzzing the discrepancy between the original data and its reconstruction. The required gradients are easily obtained by using the chain rule to backpropagate erro derivatives first through the decoder network and then theough encoder network. The whole system is called an autoencoder and

For training the Autoencoder model, we use triplet loss to optimize the model performance. The traditonal logistic loss aims to identify the relationships between the given image and the similar images optimally. However, it cannot utilize the relationship between three instances: examplar, relevant instance, and non-relevant instance. Therefore, triplet loss can learn effectively the complex features between these three instances by maximizing the similarity scores between examplar-positive pairs and minimizing those figures between exemplar-negative pairs. Therefore, the network can suggest the similar images more actively.

### Ultimate Judgement

### Evaluation

**References**

[1] [2] [3] S. Ioffe and C. Szegedy, “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”, Mar. 2015. doi: 10.48550/arXiv.1502.03167.

[4] [5] N. Srivastava, G. Hinton, A. Krizhevsky, L. Sutskever, and R. Salakhutdinov, “Dropout: a simple way to prevent neural networks from overfitting”, vol 15, no. 1, pp. 1929-1958, 2014. [Online]. Available: <https://www.jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf?utm_content=buffer79b43&utm_medium=social&utm_source=twitter.com&utm_campaign=buffer>

[6] [7] I. Loshchilova and F. Hutter, “Decoupled weight decay regularization”, 2017, doi: 10.48550/arXiv.1711.05101

[8] [9] D. Hendrycks and K. Gimpel, “Gaussian Error Linear Units (GELUs)”, 2016, doi: 10.48550/arXiv.1606.08415

[10] [11] K. He, X. Zhang, S. Ren, and J. Sun, “Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification”, 2015, doi: 10.48550/arXiv.1502.01852