Graphical user interface, text, application

Description automatically generated

**ASSESSMENT 2 - GROUP PROJECT**

**Date: 19/05/2023**

**Course:  COSC2753**

**Course Coordinator: Nguyen Thien Bao (**[**bao.nguyenthien@rmit.edu.vn**](mailto:bao.nguyenthien@rmit.edu.vn)**)**

**Lecturer: Tran Nhat Quang (**[**quang.tran26@rmit.edu.vn**](mailto:quang.tran26@rmit.edu.vn)**)**

**Group T4-G03:**

**Ngo Quoc Binh (**[**s3927469@rmit.edu.vn**](mailto:s3927469@rmit.edu.vn)**)**

**Nguyen Dang Nhat (**[**s3878292@rmit.edu.vn**](mailto:s3878292@rmit.edu.vn)**)**

**Nguyen Ngoc Luong (**[**s3927460@rmit.edu.vn**](mailto:s3927469@rmit.edu.vn)**)**

**Tran Le Bao Ngoc (**[**s3817981@rmit.edu.vn**](mailto:s3927469@rmit.edu.vn)**)**

**Mai Chieu Thuy (**[**s3877746@rmit.edu.vn**](mailto:s3877746@rmit.edu.vn)**)**

Contents

[Exploratory Data Analysis 4](#_Toc135382308)

[Data Overview 4](#_Toc135382309)

[Loading and Viewing the Data 4](#_Toc135382310)

[Key Observations 4](#_Toc135382311)

[Key Conclusions 4](#_Toc135382312)

[Task 1: Classify Flower Images 4](#_Toc135382313)

[Evaluation Framework 4](#_Toc135382314)

[Strategy 4](#_Toc135382315)

[Configurations 4](#_Toc135382316)

[Architectures 5](#_Toc135382317)

[AlexNet [23] 5](#_Toc135382318)

[VGG [27] 5](#_Toc135382319)

[MobileNetV2 [28] 5](#_Toc135382320)

[EfficientNetV2 [10] 5](#_Toc135382321)

[DarkNet53 [31] 6](#_Toc135382322)

[CSPDarkNet53 [33] 6](#_Toc135382323)

[Judgement 6](#_Toc135382324)

[Evaluation 6](#_Toc135382325)

[Task 2: Recommend Flower Images 7](#_Toc135382326)

[Evaluation Framework 7](#_Toc135382327)

[Architecture 7](#_Toc135382328)

[Extractor as Convolutional Layers in CNN Classifier 7](#_Toc135382329)

[Extractor as Encoder in Autoencoder 7](#_Toc135382330)

[Ultimate Judgement 8](#_Toc135382331)

[Evaluation 8](#_Toc135382332)

[Future work 8](#_Toc135382333)

[References 8](#_Toc135382334)

[Appendices 12](#_Toc135382335)

Tables of Figures

[Figure 1 CBIR System Architecture 7](#_Toc135382336)

[Figure 2 The distribution of classes in the given dataset 12](#_Toc135382337)

[Figure 3 The distribution of classes in the new dataset 12](#_Toc135382338)

[Figure 4 The performance results of models on task 1 12](#_Toc135382339)

[Figure 5 The recommendation results from Darknet-53 12](#_Toc135382340)

[Figure 6 The evaluation performance of all models for task 2 12](#_Toc135382341)

[Figure 7 Darknet-53 performance after first 25 epochs 13](#_Toc135382342)

[Figure 8 Darknet-53 performance saturates after about 100 epochs on the "hard augmentation" 13](#_Toc135382343)

[Figure 9 Darknet-53 accuracy reaches its peak at 78.1% 13](#_Toc135382344)

# Exploratory Data Analysis

## Data Overview

There are more than 4500 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 [1].

## Loading and Viewing the Data

Figure 2 highlights two critical issues in our dataset: class imbalance and insufficient sample size. To improve dataset generalization, it is recommended to have approximately 1000 images per class for training deep models [2]. Our goal was to generate around 1500 images for each flower type. We addressed this by utilizing the Selenium package [3] for scraping data from popular flower shops, proving to be an effective method for acquiring more images. However, the number of images obtained through scraping, over 200 per class, fell short of our target. To augment the existing images, we employed the Augmentations package [4], which transformed the input images using techniques such as shifting, scaling, saturating, color changing, and more. We applied advanced techniques such as rain, snow, and fog to introduce variations. Initially, the dataset only contained images of flowers in normal conditions, but we aimed to increase the model's generalization by including diverse scenarios. As a result, our data set now contains over 11000 images. Figure 3 summaries the distribution of new dataset. The augmentation process not only increases the sample count but also introduces images captured under different conditions. This diversification enhances the model's ability to generalize and perform well in real-life scenarios.

## Key Observations

* From further examination, we conclude that there are zero duplications in the dataset.
* The average image size is 250x250 pixels, with most around a 1:1 aspect ratio. All images are in .jpg format.

## Key Conclusions

* Monitoring the model for any remaining data quality or diversity issues will still be important to maximize performance. Additional augmentation/oversampling may be useful.
* Standardizing image sizes and formats will facilitate training and improve model generalizability.
* No major data cleaning or resampling appears needed with the final balanced and diverse dataset. The data can be used to proceed to model building and hyperparameter tuning.

In summary, a balanced and diverse dataset suitable for deep learning was obtained through web scraping and augmentation. Model training can commence with care taken to monitor for any further data requirements or issues and a need to maximize generalizability. Initial data quality checking and cleaning validated the dataset for use in building a robust model.

# Task 1: Classify Flower Images

## Evaluation Framework

We choose accuracy [5] as the main metric to evaluate model performance for three reasons. First, the issue of imbalanced datasets has been addressed, reducing the likelihood of a model achieving high accuracy through overfitting a specific class. Second, we do not prioritize the importance of a particular flower type, so recall or precision [6] is not as crucial as accuracy. However, in medical-based problems with imbalanced datasets and recognized importance of a specific class, recall or precision would be given higher priority than accuracy [7]. Finally, accuracy is the primary metric for evaluating the Flower-102 dataset [8], an official dataset for assessing a new architecture.

## Strategy

We adopt Karpathy's training strategy: baseline models, overfitting, and regularization [9]. Our approach involves overfitting a specific model on the dataset until it achieves a training accuracy of more than 90%. This method requires training the model on an uncomplicated dataset without any augmentations, allowing it to quickly attain high accuracy compared to using image augmentation techniques. By training on a simpler dataset, the model learns important characteristics, enabling it to converge more rapidly when encountering more challenging datasets through the application of image augmentation. This concept is akin to progressive learning, as described by Tan and Le [10]. However, due to limitations, we can only enhance the level of augmentation and are unable to increase the size of the input image.

## Configurations

**Batch Normalization [11]:** Internal covariate shift slows down and makes it harder to train models. To normalize layers input, we apply Batch Normalization. It acts as a regularizer and helps models to achieve the same accuracy with fewer training steps, hence, saving time as well.

**Dropout [12]:** When the data is limited, our model can be overfitted. The dropout technique is introduced to address this problem. However, dropout will increase your training time so in other words, applying dropout also means creating a trade-off between accuracy and training time.

**AdamW [13]:** Adam with L2 regularization had been proved to have worse generalization performance than AdamW. The paper also mentioned that AdamW had yielded better training loss and test error than Adam.    
**SGD [14]:** We employ SGD as the optimizer when the model performance saturates due to its superior generalization performance compared to Adam. SGD's advantage stems from its tendency to have less stabilization at local minima. This characteristic enables it to explore and potentially move to other minima that yield better results, thereby enhancing the model's generalization ability [15].

**GELU [16]:** GELUs, a smooth activation function, offers advantages over ReLUs [17] in neural networks. Its smoothness ensures continuous gradients, leading to smoother optimization landscapes and improved convergence during training. Additionally, GELUs reduces the occurrence of "dead neurons" commonly associated with ReLUs, where neurons become non-responsive due to negative inputs, resulting in gradient stagnation. GELUs non-zero gradient for all inputs mitigates this issue, potentially enhancing the network's capacity. Moreover, GELUs demonstrates better signal propagation properties, facilitated by its smoothness, enabling improved information flow, gradient propagation, and enhanced learning in deep neural networks.

## Architectures

For task 1, we exclusively utilize convolutional neural networks (CNNs) [18] due to their superior performance compared to traditional machine learning algorithms [19] and basic multilayer perceptron’s [20]. Conventional approaches treat individual image features independently, which fails to capture the underlying structural information and subsequently leads to decreased evaluation performance. In contrast, CNNs leverage convolutional operations to transform images into higher-dimensional spaces, facilitating more effective data representation and learning. While traditional methods like PCA [21] or SVD [22] can reduce image dimensionality, they are not as competitive as convolutional operations, which benefit from the optimization learning procedure embedded within CNN architectures.

### AlexNet [23]

AlexNet, the first ILSVRC [24] winner, is a groundbreaking convolutional neural network architecture that changed the field of deep learning. AlexNet offers several advantages, including direct image input into its classification model and the utilization of well-known techniques like dropout, GPU processing, parallelization, and ReLU. Furthermore, convolution layers can extract picture edges automatically, and fully connected layers can learn these features. However, compared to newer models like ResNet [25], GoogLeNet [26], and VGG [27], AlexNet has a shallower architecture, and the use of larger convolution filters (5x5) has been discouraged.

### VGG [27]

VGG is a multi-layered deep CNN architecture that utilizes small 3×3 convolution kernels to boost network depth. VGG has advantages over AlexNet since its small kernel size reduces the number of parameters and complexity significantly, resulting in faster learning. Moreover, VGG incorporates 1×1 convolution layers to enhance non-linearity features and make reliable decisions. VGG-11, with its 11 weighted layers, including convolutional and fully connected layers, allows for learning more complex features. However, drawbacks include longer training time and high computational cost due to the model's depth and fully connected nodes.

### MobileNetV2 [28]

MobileNetV2 is a lightweight deep neural network architecture that utilizes depth-wise separable convolutions [29] to reduce inference time compared to traditional convolutions. This approach replaces the traditional convolution with two independent operations: depth-wise convolution and pointwise convolution. The depth-wise convolution applies an identical filter to each input channel, while the pointwise convolution combines the outputs with a 1×1 convolution. This technique reduces parameters while maintaining accuracy. Moreover, an improvement of MobileNetV2 compared to the previous one is it applied the Inverted Residual, which is a short connection technique [25]. This technique helps prevent gradient vanishing, ensuring the loss value does not saturate after a few epochs. However, MobileNet-based architecture involves a trade-off between accuracy and inference time.

### EfficientNetV2 [10]

EfficientNetV2 is an advanced CNN architecture that improves upon its predecessor, EfficientNet [30]. It introduces "EfficientNetV2 Compound Scaling," which scales the network's depth, width, and resolution simultaneously, ensuring optimal performance across different computational resources. 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, capturing both low-level and high-level visual features for improved discriminative power. Advantages of EfficientNetV2 include its efficient design philosophy, achieving excellent performance with fewer parameters, and the integration of Residual Channel Attention Network (ReCAN), enhancing accuracy by focusing on informative regions while suppressing noise or irrelevant features. However, architecture has drawbacks. Training and inference times are longer due to increased depth and width. The complex architecture may be difficult to interpret. Furthermore, performance relies on the quality and diversity of the training data, limiting performance on dissimilar datasets and emphasizing the importance of data representation and transfer learning.

### DarkNet53 [31]

DarkNet53 is a convolutional neural network architecture initially introduced as the foundation of YOLOv3 in 2016. It builds upon DarkNet19 from YOLOv2 [32]. DarkNet53 gets its name from its 53 convolutional layers. Its design philosophy emphasizes simplicity and efficiency by stacking multiple layers with small filter sizes to capture complex features while minimizing computational complexity. Advantages of DarkNet53 include the inclusion of residual connections [25] that address the vanishing gradient problem and facilitate the training of deeper networks. It also employs feature fusion from multiple scales, enabling the detection of objects of varying sizes while preserving fine-grained details and global context. However, a limitation of DarkNet53 is that, despite using skip connections to capture contextual information, it may not capture long-range dependencies as effectively as some other architectures.

### CSPDarkNet53 [33]

CSPDarkNet53 is an extension of the DarkNet architecture that implements Cross Stage Partial Network (CSPNet) [34] 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 merge them through a cross-stage hierarchy. CSPDarkNet53 provides a training time advantage, allowing for faster convergence and reduced training durations compared to other architectures. However, a drawback of CSPDarkNet53 is the trade-off in accuracy, as it may not match the levels achieved by some alternative architectures, limiting its applicability.

## Judgement

Figure 4 summarizes the model performance on task 1. Due to our limited resources, we can just train at maximum 25 epochs per time. The accuracy of AlexNet saturates at 66% after 100 epochs, while VGG, a deeper model, achieves 68.8% accuracy, indicating its superiority due to its deeper architecture and ability to learn complex features. MobileNetV2 strikes a balance between accuracy and training time, resulting in a lower accuracy of 60.9%. EfficientNetV2, which enhances depth, width, and resolution, achieves an accuracy of 70.5%. Darknet-53 surpasses all other models with an accuracy of 78.1% by mitigating overfitting and leveraging residual blocks to increase depth without gradient vanishing/exploding concerns. CSPDarknet-53, a lightweight version, achieves an accuracy of 66.1% due to the trade-off created by the CSP layer. Consequently, Darknet-53 is chosen as the final model for task 1.

In the training process of Darknet-53, an accuracy of 21.3% is initially achieved after 25 epochs (figure 7). Applying "hard augmentation" significantly improves performance to 66% in the next 25 epochs. Further training with the “hard” augmented data results in an accuracy of approximately 72% (figure 8). Recognizing performance saturation, the strength of augmentation is increased ("strong augmentation"), leading to an accuracy of 78.1% after more than 100 epochs (figure 9). The correlation between performance improvement and augmentation difficulty is proved in the concept of progressive learning.

## Evaluation

The Flower-102 dataset [8] is commonly used to evaluate image classification neural networks. Models like EfficientNet-B7 [30] and EfficientNetV2-L [10] achieve an impressive accuracy of 98.8% on this dataset. These models achieve high accuracy by utilizing a relatively small learning rate, which helps them converge to better local minima during training. However, using a small learning rate often leads to longer training times. To mitigate this issue, multiple GPUs are commonly employed to accelerate the training process. In our work, we trained our models using the V100 GPU provided by Google Colab [41], which may not be as optimal as the setup of such novel experiments. Moreover, transfer learning plays a crucial role in the performance of these models [35]. Before being trained on the Flower-102 dataset, these models have already been pretrained on the large-scale ImageNet dataset [36]. This prior knowledge about the characteristics of a diverse dataset allows these models to improve their performance more efficiently compared to training from scratch on the target dataset. Transformer-based architectures, such as CCT-14/7x2 [37], have shown exceptional accuracy, reaching up to 99.76%. However, to fully exploit the power of transformers, an extremely large data set containing millions of images is often required [38]. Recently, novel training techniques have been discovered to bridge the performance gap between transformer models trained without pretraining and extensive augmentation techniques and conventional CNN architectures [39]. ResNet50, a model like Darknet-53, achieves an accuracy of 97.9% on the Flower-102 dataset by employing a special training procedure in timm [40]. However, replicating such an environment may be challenging due to hardware limitations, requiring multiple V100 GPUs and significant training hours.

# Task 2: Recommend Flower Images

## Evaluation Framework

Precision and recall rates are commonly used to evaluate Content-based Image Retrieval (CBIR) systems [42]. Precision measures the system's ability to retrieve similar images to the given input, while recall assesses the retrieval of relevant images from the entire database. More specifically, mean average precision (mAP) is the metric for evaluating model performance [43]. Accuracy is not considered in the CBIR system due to its binary classification nature, distinguishing between relevant and non-relevant items. However, accuracy can be misleading as it may classify all items as non-relevant, given the significantly larger number of non-relevant samples. This approach is ineffective for image retrieval as users always prioritize acquiring relevant information [44].

Popular metrics for image comparison include L1, L2, cosine, and correlation. L1 and L2 are not chosen as they compare images pixel by pixel without considering semantic relationships. Cosine and correlation have advantages over the former metrics, as they capture contextual relationships more efficiently and handle brightness variance. Correlation is chosen as the main method because it considers both direction and magnitude variance between images, which is crucial for measuring similarity. Brightness-varied images, generated as an augmentation method, can be more accurately captured using correlation. However, cosine may be considered if object locations are significantly shifted during augmentation, as semantic similarity becomes more important than initial 3-D space location in lower-dimensional presentations.

## Architecture

The independence of pixels in each image poses a challenge for metrics to accurately capture image characteristics. Naively applying these metrics also leads to high computational requirements due to evaluating all pixels. To address this, we introduce an extractor that reduces image dimensionality, enabling calculations on smaller data sets and reducing computational efforts. The reduced information is highly correlated, capturing image characteristics more sharply for precise similarity evaluation.

A picture containing text, screenshot, font, rectangle

Description automatically generated

Figure 1 CBIR System Architecture

Figure 1 illustrates our CBIR system architecture [45]. Initially, an extractor processes the image database, extracting and indexing features while storing the information in a local file. Pre-calculating image features minimize redundant calculations during image queries. The extractor then computes the input image's characteristics and calculates similarity scores for each image. The K-nearest neighbor algorithm [46] is applied to identify the 10 most similar images, which are subsequently ranked and returned. Two implementation approaches for the extractor are outlined below.

### Extractor as Convolutional Layers in CNN Classifier

We utilize convolutional layers from a CNN classifier as an extractor in task 1 to reduce image dimensionality and capture image features. These layers have learned dataset features through the classification task, providing insights into the dataset's distribution. This approach leverages the existing models from Task 1 to accomplish the objectives of Task 2.

### Extractor as Encoder in Autoencoder

Incorporating an autoencoder as an alternative extractor in our CBIR system provides advantages over traditional methods like PCA [21] or SVD [22]. These linear methods are unable to capture non-linear relationships among image features. Autoencoder, on the other hand, leverages the power of non-linearity in deep networks [47]. It consists of an encoder, bottleneck, and decoder, with the encoder serving as our focus. After training, the encoder becomes an extractor in our CBIR system, revealing hidden relationships between images [48]. To train the autoencoder model, we employ triplet loss [49] to optimize its performance. Traditional logistic loss [50] only considers the relationship between a given image and similar images, while triplet loss incorporates three instances: exemplar, relevant instance, and non-relevant instance. By maximizing similarity scores between exemplar-positive pairs and minimizing them between exemplar-negative pairs, triplet loss effectively learns complex features among these instances.

## Ultimate Judgement

In Figure 6, the CBIR (Content-Based Image Retrieval) systems are evaluated based on correlation distance [51] and mAP. Darknet-53 achieves the highest performance with a correlation distance of approximately 0.8, followed by EfficientNetV2 with a mAP of around 0.7. AlexNet surpasses VGG and CSPDarkNet-53 to secure the third position with a mAP exceeding 0.6. Despite CSPDarknet-53 having the lowest correlation distance (approximately 0.018) among the models, its overall performance is poor. The autoencoder model disappoints with a mAP of only 0.38, likely due to ineffective dimensionality reduction of the extracted features, hindering its ability to learn from high-quality data and make reliable decisions. Figure 5 shows the recommendations from Darknet-53.

## Evaluation

Currently, there are no published studies on our specific flower recommendation system. However, CBIR systems have been successfully applied in various fields. For example, in person re-identification, the UED approach [52] achieves a mAP of up to 0.928 on the Market1501 dataset [53] by incorporating the strengths of different diffusion methods while mitigating their limitations. Although the UED approach involves non-convex optimization, which makes finding the best minimum more challenging, its careful architecture formulation and derivation contribute to its performance. Unfortunately, due to limited resources, we cannot implement these methods in our work. In remote sensing image retrieval, a system utilizing triplet loss achieves an impressive mAP of 0.9663 [54]. Triplet loss effectively captures the relationship between three objects: an exemplar, a relevant object, and a non-relevant object. Additionally, the model performance is enhanced through a combination of supervised learning (fully connected layer of the CNN) and unsupervised learning (PCA) for dimensionality reduction.

# Future work

**Enploying more domain knowledge:** Our model experiences misclassification due to unclear labels in the dataset. Specifically, the plants Ping Pong [57], Rosy [58], and Tana [59] belong to the Chrysanthemum genus [60], resulting in significant misclassifications. By gaining a deeper understanding of plant genera and providing more accurate labels for such images, we can improve the quality of our dataset.

**Implementing Object Detection Methods for extracting flowers:** The model's decisions are significantly influenced by the image background, resulting in some images being classified or suggested based on the objects in the background rather than the flowers themselves. By implementing object detection techniques like Faster R-CNN [55] and Mask R-CNN [56], we can mitigate the impact of the background and improve the model's performance.

**Applying GAN-based methods for advanced augmentation:** In our work, we have added advanced weather conditions for regularizing our model. However, by implementing some generative adversarial network (GAN) techniques [61] such as CycleGAN [62], Pix2pix [63], DCGAN [64], etc., a myriad number of images can be generated to enrich the generalization of our dataset [65].

**Implementing SOTA methods:** In the evaluation section, various state-of-the-art methods have demonstrated their effectiveness in achieving high-performance results. For task 1, we will explore advanced techniques such as Vision Transformers [38] and Graph Convolution Networks [68] to cover a wide range of approaches. Additionally, we will incorporate advanced training technologies like training in timm [67] and applying sharpness-aware minimizer (SAM) [39]. In task 2, we will investigate the use of autoencoder architecture [48] to enhance performance and integrate PCA-related techniques [54] to optimize the feature extraction process. Furthermore, we will explore the impact of image characteristics such as color, shape, and texture on improving the performance of CBIR systems.

**Equiping more hardware resources:** The power of GPU has been recognized in enhancing the learning procedure [66]. Therefore, we can increase the boost the learning time and try more models with a huge number of advanced hardware components.

# References

[1]V. R. Joseph, “Optimal ratio for data splitting,” *Statistical Analysis and Data Mining: The ASA Data Science Journal*, vol. 15, no. 4, pp. 531–538, Apr. 2022, doi: <https://doi.org/10.1002/sam.11583>.

[2]S. Shahinfar, P. Meek, and G. Falzon, “How many images do I need? Understanding how sample size per class affects deep learning model performance metrics for balanced designs in autonomous wildlife monitoring,” *arXiv.org*, Oct. 16, 2020. <https://arxiv.org/abs/2010.08186> (accessed May 19, 2023).

[3]“SeleniumHQ Browser Automation,” *www.selenium.dev*. <https://www.selenium.dev/>

[4]“Albumentations,” *albumentations.ai*. <https://albumentations.ai/>

[5]NGSS Science and Engineering Practices, “Practices of Science: Precision vs. Accuracy | manoa.hawaii.edu/ExploringOurFluidEarth,” *Hawaii.edu*, 2020. <https://manoa.hawaii.edu/exploringourfluidearth/physical/world-ocean/map-distortion/practices-science-precision-vs-accuracy>

[6]J. Davis and M. Goadrich, “The Relationship Between Precision-Recall and ROC Curves.” Available: <https://www.biostat.wisc.edu/~page/rocpr.pdf>

[7]S. A. Hicks *et al.*, “On evaluation metrics for medical applications of artificial intelligence,” *Scientific Reports*, vol. 12, no. 1, p. 5979, Apr. 2022, doi: <https://doi.org/10.1038/s41598-022-09954-8>.

[8]“Visual Geometry Group - University of Oxford,” *www.robots.ox.ac.uk*. <https://www.robots.ox.ac.uk/~vgg/data/flowers/102/>

[9]“A Recipe for Training Neural Networks,” *karpathy.github.io*. <http://karpathy.github.io/2019/04/25/recipe/>

[10]M. Tan and Q. V. Le, “EfficientNetV2: Smaller Models and Faster Training,” *arXiv:2104.00298 [cs]*, Jun. 2021, Available: <https://arxiv.org/abs/2104.00298>

[11]S. Ioffe and C. Szegedy, “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift,” *arXiv.org*, 2015. <https://arxiv.org/abs/1502.03167>

[12]N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A Simple Way to Prevent Neural Networks from Overfitting,” *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014, Available: <https://jmlr.org/papers/v15/srivastava14a.html>

[13]I. Loshchilov and F. Hutter, “Decoupled Weight Decay Regularization,” *arXiv.org*, Jan. 04, 2019. <https://arxiv.org/abs/1711.05101v3> (accessed May 19, 2023).

[14]S. Ruder, “An overview of gradient descent optimization algorithms,” *arXiv.org*, 2016. <https://arxiv.org/abs/1609.04747>

[15]P. Zhou, J. Feng, C. Ma, C. Xiong, S. Hoi, and W. E, “Towards Theoretically Understanding Why SGD Generalizes Better Than ADAM in Deep Learning,” *arXiv:2010.05627 [cs, math, stat]*, Nov. 2021, Available: <https://arxiv.org/abs/2010.05627>

[16]D. Hendrycks and K. Gimpel, “Gaussian Error Linear Units (GELUs),” *arXiv:1606.08415 [cs]*, Jul. 2020, Available: <https://arxiv.org/abs/1606.08415>

[17]Agarap, Abien Fred, “Deep Learning using Rectified Linear Units (ReLU),” *arXiv.org*, 2018. <https://arxiv.org/abs/1803.08375>

[18]K. O’Shea and R. Nash, “An Introduction to Convolutional Neural Networks,” *arXiv:1511.08458 [cs]*, Dec. 2015, Available: <https://arxiv.org/abs/1511.08458>

[19]S. Ray, “A Quick Review of Machine Learning Algorithms,” *IEEE Xplore*, Feb. 01, 2019. <https://ieeexplore.ieee.org/document/8862451>

[20]A. Rana, A. Singh Rawat, A. Bijalwan, and H. Bahuguna, “Application of Multi Layer (Perceptron) Artificial Neural Network in the Diagnosis System: A Systematic Review,” *IEEE Xplore*, Aug. 01, 2018. <https://ieeexplore.ieee.org/abstract/document/8509069> (accessed May 13, 2023).

[21]“Lesson 11: Principal Components Analysis (PCA) | STAT 505,” *PennState: Statistics Online Courses*. <https://online.stat.psu.edu/stat505/lesson/11>

[22]“Singular Value Decomposition (SVD) tutorial,” *web.mit.edu*. <https://web.mit.edu/be.400/www/SVD/Singular_Value_Decomposition.htm>

[23]A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” *Neural Information Processing Systems*, 2012. <https://papers.nips.cc/paper_files/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html>

[24]O. Russakovsky *et al.*, “ImageNet Large Scale Visual Recognition Challenge,” *arXiv.org*, 2014. <https://arxiv.org/abs/1409.0575>

[25]K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” *arXiv.org*, Dec. 10, 2015. <https://arxiv.org/abs/1512.03385>

[26]C. Szegedy *et al.*, “Going Deeper with Convolutions,” *arXiv.org*, 2014. <https://arxiv.org/abs/1409.4842>

[27]K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” *arXiv.org*, 2014. <https://arxiv.org/abs/1409.1556>

[28]M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “MobileNetV2: Inverted Residuals and Linear Bottlenecks,” *arXiv.org*, 2018. <https://arxiv.org/abs/1801.04381>

[29]A. G. Howard *et al.*, “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications,” *arXiv.org*, 2017. <https://arxiv.org/abs/1704.04861>

[30]M. Tan and Q. V. Le, “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks,” *arXiv.org*, 2019. <https://arxiv.org/abs/1905.11946>

[31]J. Redmon and A. Farhadi, “YOLOv3: An Incremental Improvement,” *arXiv.org*, 2018. <https://arxiv.org/abs/1804.02767>

[32]J. Redmon and A. Farhadi, “YOLO9000: Better, Faster, Stronger,” *arXiv.org*, 2016. <https://arxiv.org/abs/1612.08242>

[33]A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, “YOLOv4: Optimal Speed and Accuracy of Object Detection,” Apr. 2020, Available: <https://arxiv.org/abs/2004.10934>

[34]C.-Y. Wang, H.-Y. M. Liao, I.-H. Yeh, Y.-H. Wu, P.-Y. Chen, and J.-W. Hsieh, “CSPNet: A New Backbone that can Enhance Learning Capability of CNN,” *arXiv:1911.11929 [cs]*, Nov. 2019, Available: <https://arxiv.org/abs/1911.11929>

[35]K. Weiss, T. M. Khoshgoftaar, and D. Wang, “A survey of transfer learning,” *Journal of Big Data*, vol. 3, no. 1, May 2016, doi: <https://doi.org/10.1186/s40537-016-0043-6>.

[36]J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “ImageNet: A large-scale hierarchical image database,” *2009 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2009, doi: <https://doi.org/10.1109/cvpr.2009.5206848>.

[37]A. Hassani, S. Walton, N. Shah, A. Abuduweili, J. Li, and H. Shi, “Escaping the Big Data Paradigm with Compact Transformers,” *arXiv:2104.05704 [cs]*, Aug. 2021, Available: <https://arxiv.org/abs/2104.05704>

[38]A. Dosovitskiy *et al.*, “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale,” *arXiv:2010.11929 [cs]*, Oct. 2020, Available: <https://arxiv.org/abs/2010.11929>

[39]X. Chen, C.-J. Hsieh, and B. Gong, “When Vision Transformers Outperform ResNets without Pre-training or Strong Data Augmentations,” *arXiv.org*, Mar. 13, 2022. <https://arxiv.org/abs/2106.01548> (accessed May 19, 2023).

[40]R. Wightman, H. Touvron, and H. Jégou, “ResNet strikes back: An improved training procedure in timm,” *arXiv.org*, Oct. 01, 2021. <https://arxiv.org/abs/2110.00476> (accessed May 19, 2023).

[41]Google, “Google Colaboratory,” *Google.com*, 2019. <https://colab.research.google.com/>

[42]H. Müller, W. Müller, D. McG. Squire, S. Marchand-Maillet, and T. Pun, “Performance evaluation in content-based image retrieval: overview and proposals,” *Pattern Recognition Letters*, vol. 22, no. 5, pp. 593–601, Apr. 2001, doi: <https://doi.org/10.1016/s0167-8655(00)00118-5>.

[43]“Section 8 (Week 8),” *cs230.stanford.edu*. <https://cs230.stanford.edu/section/8/>

[44]“Evaluation in information retrieval,” *nlp.stanford.edu*. <https://nlp.stanford.edu/IR-book/html/htmledition/evaluation-in-information-retrieval-1.html>

[45]X. Li, J. Yang, and J. Ma, “Recent developments of content-based image retrieval (CBIR),” *Neurocomputing*, vol. 452, pp. 675–689, Sep. 2021, doi: <https://doi.org/10.1016/j.neucom.2020.07.139>.

[46]O. Kramer, “K-Nearest Neighbors,” *Dimensionality Reduction with Unsupervised Nearest Neighbors*, vol. 51, pp. 13–23, 2013, doi: <https://doi.org/10.1007/978-3-642-38652-7_2>.

[47]G. E. Hinton, “Reducing the Dimensionality of Data with Neural Networks,” *Science*, vol. 313, no. 5786, pp. 504–507, Jul. 2006, doi: <https://doi.org/10.1126/science.1127647>.

[48]A. Krizhevsky and G. E. Hinton, “Using very deep autoencoders for content-based image retrieval,” *Semantic Scholar*, 2011. <https://www.semanticscholar.org/paper/Using-very-deep-autoencoders-for-content-based-Krizhevsky-Hinton/88080d28536f36588740737f3b7a1f6c1a409654> (accessed May 19, 2023).

[49]E. Hoffer and N. Ailon, “Deep metric learning using Triplet network,” *arXiv:1412.6622 [cs, stat]*, Dec. 2018, Available: <https://arxiv.org/abs/1412.6622>

[50]J. Daniel and J. Martin, “Speech and Language Processing.” Available: <https://web.stanford.edu/~jurafsky/slp3/5.pdf>

[51]“scipy.spatial.distance.correlation — SciPy v1.10.1 Manual,” *docs.scipy.org*. <https://docs.scipy.org/doc/scipy/reference/generated/scipy.spatial.distance.correlation.html> (accessed May 19, 2023).

[52]S. Bai, P. Tang, P. H. S. Torr, and L. J. Latecki, “Re-Ranking via Metric Fusion for Object Retrieval and Person Re-Identification,” *IEEE Xplore*, Jun. 01, 2019. <https://ieeexplore.ieee.org/document/8953719> (accessed May 19, 2023).

[53]L. Zheng, L. Shen, L. Tian, S. Wang, J. Wang, and Q. Tian, “Scalable Person Re-identification: A Benchmark,” *IEEE Xplore*, Dec. 01, 2015. <https://ieeexplore.ieee.org/document/7410490>

[54]R. Cao *et al.*, “Enhancing remote sensing image retrieval using a triplet deep metric learning network,” *International Journal of Remote Sensing*, vol. 41, no. 2, pp. 740–751, Aug. 2019, doi: <https://doi.org/10.1080/2150704x.2019.1647368>.

[55]S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” *arXiv.org*, 2015. <https://arxiv.org/abs/1506.01497>

[56]K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask R-CNN,” *arXiv.org*, 2017. <https://arxiv.org/abs/1703.06870>

[57]C. Wang *et al.*, “Natural compounds with xanthine oxidase inhibitory activity: A review,” vol. 93, no. 4, pp. 387–418, Jan. 2019, doi: <https://doi.org/10.1111/cbdd.13437>.

[58]“Chrysanthemum hybridization in 21st century – A review,” *Journal of Ornamental Horticulture*, vol. 23, no. 2, 2020, Accessed: May 19, 2023. [Online]. Available: <https://www.indianjournals.com/ijor.aspx?target=ijor:joh&volume=23&issue=2&article=001>

[59]G. Mathys and E. A. Baker, “An Appraisal of the Effectiveness of Quarantines,” *Annual Review of Phytopathology*, vol. 18, no. 1, pp. 85–101, Sep. 1980, doi: <https://doi.org/10.1146/annurev.py.18.090180.000505>.

[60]G. R. Rout and P. Das, “Recent trends in the biotechnology of Chrysanthemum: a critical review,” *Scientia Horticulturae*, vol. 69, no. 3–4, pp. 239–257, May 1997, doi: <https://doi.org/10.1016/s0304-4238(97)00008-3>.

[61]I. J. Goodfellow *et al.*, “Generative Adversarial Networks,” *arXiv.org*, 2014. <https://arxiv.org/abs/1406.2661>

[62]J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks,” *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct. 2017, doi: <https://doi.org/10.1109/iccv.2017.244>.

[63]P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-Image Translation with Conditional Adversarial Networks,” *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul. 2017, doi: <https://doi.org/10.1109/cvpr.2017.632>.

[64]A. Radford, L. Metz, and S. Chintala, “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks,” *arXiv.org*, 2015. <https://arxiv.org/abs/1511.06434>

[65]Y. Lu, D. Chen, E. Olaniyi, and Y. Huang, “Generative adversarial networks (GANs) for image augmentation in agriculture: A systematic review,” *Computers and Electronics in Agriculture*, vol. 200, p. 107208, Sep. 2022, doi: <https://doi.org/10.1016/j.compag.2022.107208>.

[66]S. Mittal and S. Vaishay, “A survey of techniques for optimizing deep learning on GPUs,” *Journal of Systems Architecture*, vol. 99, p. 101635, Oct. 2019, doi: <https://doi.org/10.1016/j.sysarc.2019.101635>.

[67]“Pytorch Image Models (timm) | timmdocs,” *timm.fast.ai*. <https://timm.fast.ai/>

[68]T. N. Kipf and M. Welling, “Semi-Supervised Classification with Graph Convolutional Networks,” *arXiv:1609.02907 [cs, stat]*, Feb. 2017, Available: <https://arxiv.org/abs/1609.02907>

# Appendices

A picture containing text, screenshot, diagram, rectangle

Description automatically generated

Figure The distribution of classes in the given dataset

A picture containing text, screenshot, diagram, font

Description automatically generated

Figure The distribution of classes in the new dataset

A picture containing text, screenshot, diagram, rectangle

Description automatically generated

Figure The performance results of models on task 1

A picture containing vase, flowerpot, floral design, flower arranging

Description automatically generated

Figure The recommendation results from Darknet-53

A picture containing text, screenshot, diagram, plot

Description automatically generated

Figure The evaluation performance of all models for task 2

A graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of

Description automatically generated with low confidence

Figure Darknet-53 performance after first 25 epochs

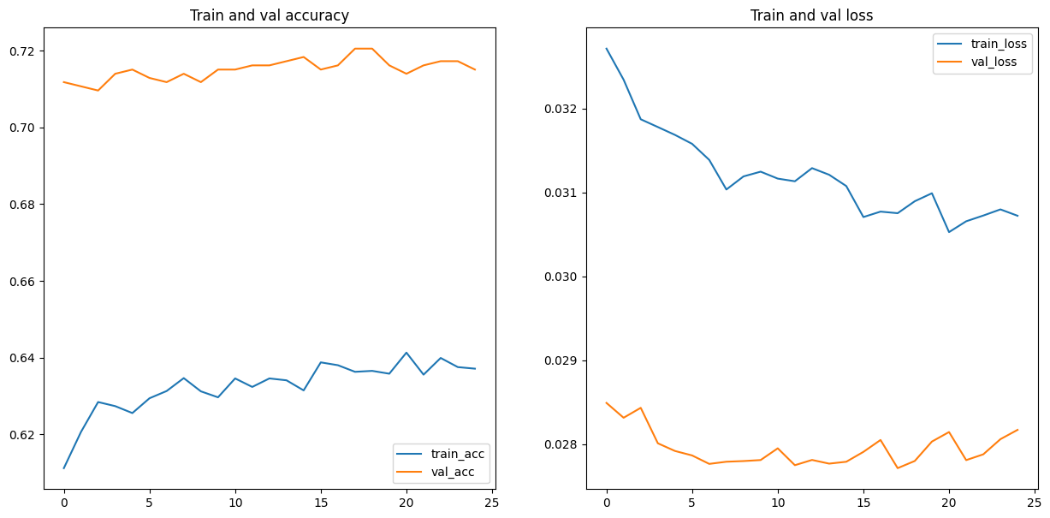


Figure Darknet-53 performance saturates after about 100 epochs on the "hard augmentation"

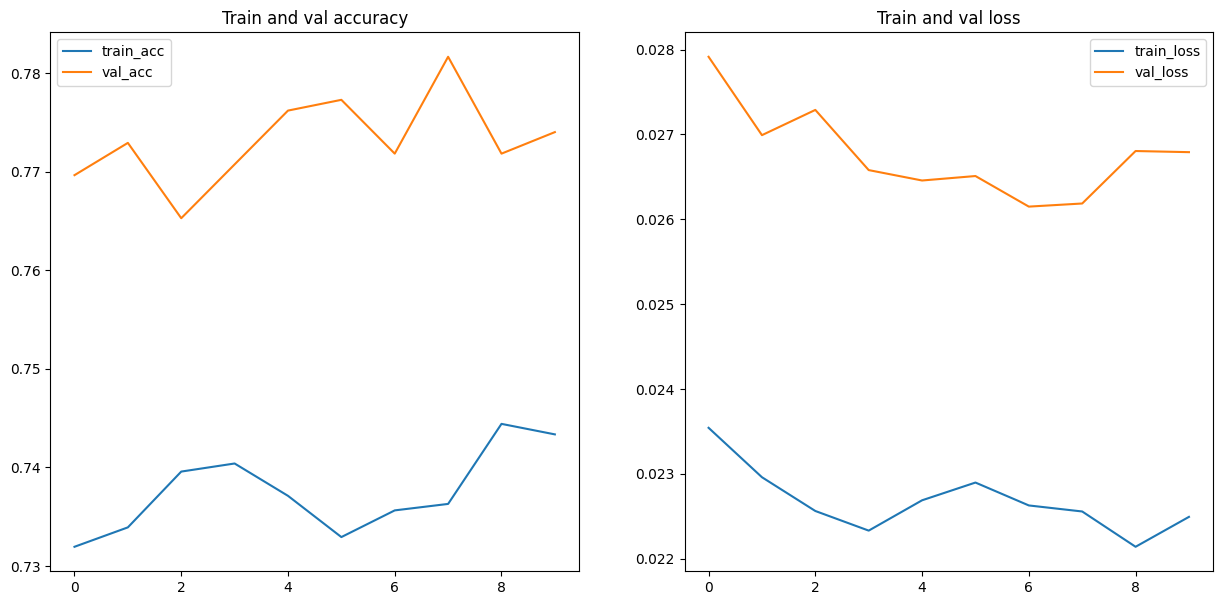


Figure Darknet-53 accuracy reaches its peak at 78.1%