Multiple Deep Learning Models Performance Comparison for Animal Classifications

Duc Nhan Do, Hong Phuc Pham
The Univeristy of Adelaide

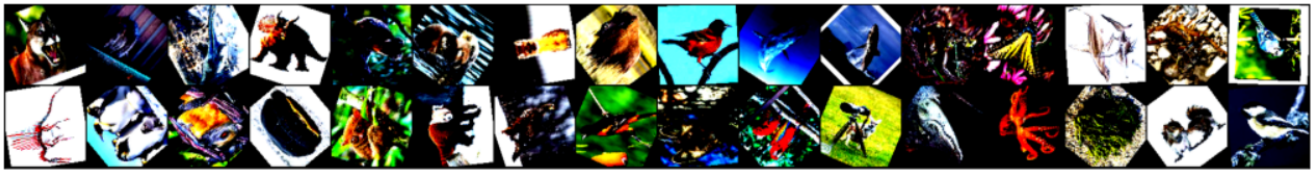
Abstract

Classification is one of the vital applications of computer vision in human life. Through the ongoing development of technologies and the colossal contribution of several research projects, intelligent, various pretrained deep learning models have been developed and improved through the years. Although new updated or introduced models have resolved their predecessors' problems and achieved higher accuracy, the classification is still a difficult task and animal classification is specific due to many visual representations in the scenes. This paper will go through some remarkable models, VGG16 [7], Inceptionv3 [8], ResNet50 [1] and EfficientNet [9]. The dataset used for this investigation is online. It contains different animals and is broken into three subsets for training, validating and testing. Tricks and tweaks such as transfer learning, altering training parameter, data augmentation, and architectural change are also used to push the chosen model to achieve higher accuracy. The result of the best model reached 99.66%, with 0.21G for floating point operation per second (FLOPs) [3]. This accuracy score shows a positive reflection of current achievement in overall animal classification.

1. Introduction

Computer vision (CV) applications have taken place in many modern practices and fields such as autonomy, health, imagery, and games [4]. The demand for implementing CV into practice solutions is growing over time. Classification is one of the main tasks in CV, drawing a lot of researchers' attention. This task involves categorising and giving labels to vectors or collections of image pixels of a picture based on particular rules due to the stream of technology evolution and several queries about upgrading the quality of life and curiosity to unlock universal mysteries. The high requirements from the public generate a competitive race between discoveries and innovation offers. Increasingly, many researchers propose to public superior to previous introductions. This contention is no longer about tens of accuracy numbers; they chase each other and overcome every single unit for accuracy. Furthermore, the computation factors are another concern of both users and researchers. That is why they are also the must-have criteria when technical proposals are put on the scale. Recently published models provide tremendous results not only in the performance but also in the resources used [5].

Animal classification is one of the regular appliances of CV in research, model measurements and comparison, for the study investigation from the expert researchers, engineers to the discipline. Although it is in the classification suit, the animal still needs a different approach from other objects like humans, plants, vehicles, tools, and furniture. In terms of animals, there are several types that could be specifically broken down into species. They are born with identical traits, unique characteristics of their body shape, fur, feather, skin type, and colour [10]. Some characteristics can be quickly recognised and identified, but some could be complex and can not be distinguished with naked eyes. On the biological level, objects need to be observed through different conditions and overtimes to capture their transformation, such as their adaptation to weather change and their body over time. Back to computing, those species presentations would be extended to what we think, and logical reasoning is different from a machine. Traits and characteristics are what are called features that can be extracted from images. Adding to the features, a critical feature in differentiating animal traits and permanent features is the visual effect on the image. Numerous conditions could affect the quality of data on images, such as illuminating factors-exposure, contrast or perception. In the traditional approach with handcrafted features, many manuals and supervised works are needed for tuning, trial and adjustment to establish a data pool for features. In contrast, deep learning (DL) - a subset of machine learning, can perform many analytical or physical tasks without human intervention. These models use a convolution neural network (CNN) and turn images into feature maps with mathematical operations [7] . After AlexNet was introduced in 2012, more and more DL models were proposed and performed well with high accuracy within a certain number of parameters. A particular model will be run through training, tested with different



(a) Different animal pictures samples from the data-set.

(b) Cheetah class pictures samples from the data-set.

Figure 1. Picture showcases from dataset

conditions and applied training methods to achieve the target of accuracy over 90% and FLOPs below 0.5G. The tests are conducted to evaluate the fulfilment of models on this specific dataset. Moreover, this is also a great reflection of animal classification work in computer vision, a showcase of tricks and tweaks using and problem handling to obtain the stated target. The proposed network will be based on the *Baseline Model*. The baseline architecture uses multiple CNNs with *ReLu* activation function followed up with *MaxPooling* layers to extract image features. After that, the image features will be flattened into a one-dimensional vector and fed to a fully connected layer to classify its corresponding label.

2. Material

The provided dataset has 151 different animals with the size of 6270 images corresponding to the animal's name. Each image is in RGB format with the height and width of 224 x 224 pixels. This collection has rich content with different scales and rotations [Fig. 1a]. Dataset also obtains different poses, angles, illuminate conditions, occlusion, and visual effects of animals on the picture [Fig. 1b].

3. Methods

All the experiments will be run on Google Colaboratory. Dataset will be divided into three sets for training, validation and testing, where the test set takes a tenth of the data size and the validation takes 20%. Therefore train, validation, and test sets will have sizes of 5530, 313, and 627, respectively. Overalls, there are two main runs: quick test model performance to choose the best model and tuning the best model. While choosing the best model, the model will be considered on three train setups: untrained model, pretrained model with no augment, and pretrained model with augmenting models. Fundamental transformations will be applied in every dataset, which includes image resize and normalisation. Images will be scaled down to half their size - 112 x 112 pixels. The reduction in the number of image pixels will highlight the main features of an image and cut down a considerable number of computing operations which will reduce computation time. The normalisation is another crucial step to transforming all images into a specific range of data for later processing. The normalisation follows this setting:

```
transforms.Normalize(mean=[0.485,
    0.456, 0.406], std=[0.229, 0.224,
    0.225])
```

Due to the thirst for deep learning data, the more data, the better the model can learn. However, it is not easy to collect data, and images in the dataset could be inconsistent where they could be either too big or small, on different scales and in specific rotations. Moreover, over-adding or the uneven number of data on some particular classes could generate unbalanced data that is over-fitted to that classes. Therefore, extra augmented transforms are added to extend the breadth of model information, which supports the model in learning various versions of an image. Random vertical flip, random horizontal flip and random rotation are added to the transform step for training by following lines:

```
transforms.RandomVerticalFlip(),
transforms.RandomHorizontalFlip(),
transforms.RandomRotation(100),
```

In the initial test to find the suitable model, models will be trained in 10 epochs, using Adam optimiser [2] with the learning rate at 0.0001 (common learning rate for Adam). For tuning, two more fully connected layers are added to the chosen model with the expectation of gaining more accuracy through feature extraction. The best hyper-parameter for the chosen model will be tried on these params range on the first layer size after flatten, learning rate and batch size:

```
"l1": tune.choice([256, 512, 1024]),
```

```
"lr": tune.choice([1e-4, 1e-3, 1e-2]),
"batch_size": tune.choice([16, 32, 64])
```

The best model will go through train again with the pretrained weight used in the first 20 epochs to accelerate the training. As the pretrained weight will be used, only minor adjustments are needed, which is why in the initial runs, CNN layers will be frozen. Then train for another 10 epochs with unfreeze feature blocks to let them learn more about the data.

4. Result

The process follows the mentioned test and runs through these pretrained models: VGG-16, Inceptionv3, ResNet50, EfficientNet (B0, B1, B7). Besides accuracy on each dataset, the FLOPs will be considered along. The calculation for FLOPs is provided as `FLOPs_couter.py` file. The printed result will be the sum FLOPs of computing from convolution layers, linear layers, batch normalisation, ReLU activation function and pooling. In theory, FLOPs will be calculated by following rules:

Convolutions - FLOPs = 2x Number of Kernel x Kernel Shape x Output Shape

Fully Connected Layers - FLOPs = 2x Input Size x Output Size

Pooling Layers - FLOPs = Height x Depth x Width of an image

With a stride, FLOPs = (Height / Stride) x Depth x (Width / Stride) of an image

Based on the calculation above and the given code, the FLOPs for each intended model are: 7.67 GFLOPs for VGG16 model, 1.07 GFLOPs for InceptionV3 model, ResNet with 2.15 GFLOPs, EffiecientNet group where B0 model got 0.21 GFLOPs, the value on B1 model is 2.88 GFLOPs, and B7 one reaches 0.31 GFLOPs. The raw conclusion can be given based on this early observation as the VGG16 seems to consume most computing resources in this case. The higher version of the model in the EfficientNet family, the more resources they require. The top-3 match is used in evaluating the model.

4.1. Initial review

Based on the comparison conducted on dataset CIFAR-100, Flowers, and other three transfer learning datasets, EffiencientNet-B7 gained 84.3% top-1 accuracy on ImageNet in 2020 [9]. () showed accuracy gain differently between models, correspondingly increasing start with Inception, ResNet, and EfficientNet. Through that info, we could expect that EffiencientNet might get the best performance, then ResNet, along with Inception and VGG. InceptionV3 got a fair accuracy above 50%, and EfficientNet generally worked well.

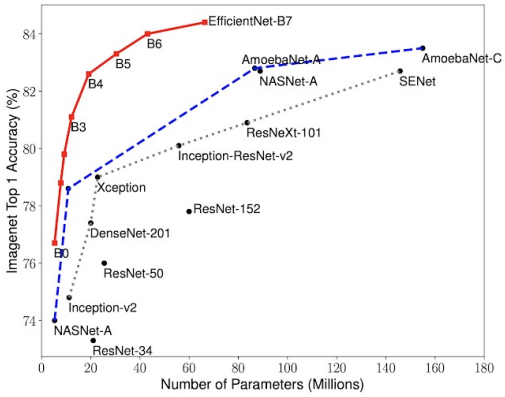


Figure 2. Models comparison

4.2. No pretrained weight with normal transformation

In the ten initial runs with no pretrained weights, while using the model architecture from remarkable models, the VGG16 model gained the best accuracy on the test set with 98.15%. It learnt quickly in the first seven epochs and then slowed down later. ResNet learnt gradually and got the second best accuracy in this test with 72.74%. In the last two training epochs, ResNet seems to have reached its bottleneck. Three models have a low rate below 35%. They might need more training to learn more about this dataset.

4.3. Pretrained weight with normal transformation

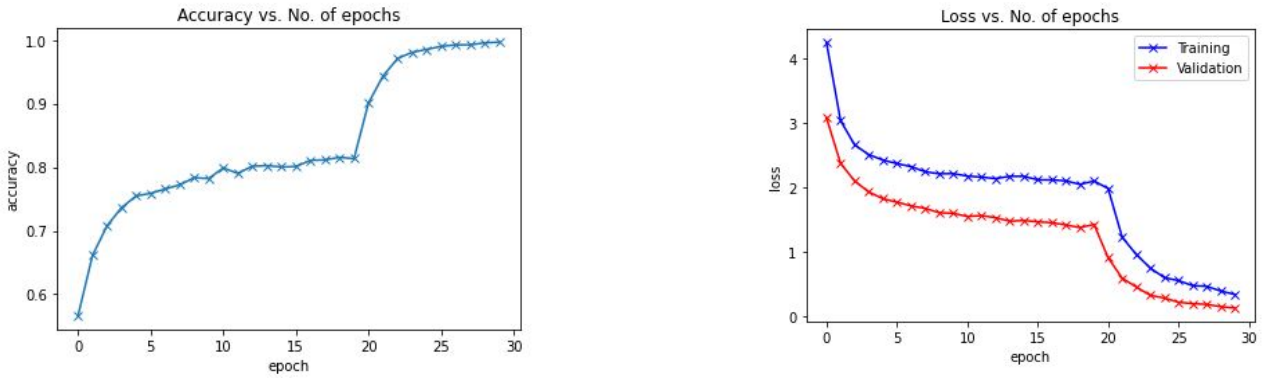
The model with pretrained weights was quickly applied and got decent accuracy. Every model has an accuracy of over 50%. Only VGG16 model reaches 100% accuracy. ResNet50 and EfficientNet B1 got over 90% where 95.37% for ResNet and 92.48% is what EfficientNet B1 achieved. EfficientNet B7 was expected to get the highest accuracy for this dataset, but in this trial, this model only made it to 53.17% - the lowest accuracy compared to others. The accuracy of these models implies a particular application of them to different datasets. At accuracy over than 80% they could be said to be acceptable performance.

4.4. Pretrained weight with extra transformation

On this run, VGG16 still get the highest performance, with 92.70%. Coming after the VGG16 model is ResNet with 81.20%. EfficientNet model family got around about 45% to 70%. One more time, the EfficientNetB7 model is the one that gets the lowest accuracy among the three. InceptionV3's rate drops dramatically to 2.53%, which is the lowest performance in this trial. The extra augmentation has given the model more data to learn. It could be why the rate

	VGG16	InceptionV3	ResNet50	EfficientNet B0	EfficientNet B1	EfficientNet B7
Trial 1	98.15%	59.69%	72.74%	5.08%	31.37%	21.86%
Trial 2	100.0%	69.98%	95.37%	89.52%	92.48%	53.17%
Trial 3	92.70%	2.53%	81.20%	70.18%	60.15%	45.18%
FLOPs	7.67G	1.07G	2.15G	0.21G	0.31G	2.88G

Figure 3. Model accuracy through trials and model FLOP. **Trial 1** - No pretrained weight with normal transformation, **Trial 2** - Pretrained weight with normal transformation, **Trial 3** - Pretrained weight with extra transformation



(a) Accuracy on validation dataset.

(b) Loss on train and validation dataset.

Figure 4. Best model validation accuracy with Loss on train and validation sets

dropped in this trial since there are more images for models to learn in the first ten epochs. However, more training or tuning could help them gain back higher accuracy.

4.5. Tuning hyper parameters

VGG16 has worked well in three trials. The accuracy of this model is stable and be kept over ninety. However, it consumes a decent amount of computing resources and is the highest FLOPs among the six models. ResNet50 is the second model that got stable accuracy. Although its FLOPs is a third of VGG16, it is still ways above the expected FLOPs. InceptionV3 is unpredictable performance in this case. In the third trial, this model's accuracy declined, and validation loss was much higher than training loss (when it keeps the same values over time). InceptionV3 got overfitted in the third run. In the EfficientNet family, FLOPs of B0 and B1 are low and meet our initial target. B0 can be chosen for further study as it has the smallest value of six of them. Its accuracy before and after applying extra transformation is not highly contrasted. Their accuracy is acceptable and may get higher after tuning.

After tuning, the hyper-parameter values returned are 512 nodes for layer one and 0.001 for the learning rate with a batch size of 64. At this state, the base accuracy reaches 79.47%. After the first 20 train epochs with the freeze feature block, the model learns a bit and enhances the accuracy to 81.33%. In the next training ten epoch with unfreeze feature blocks, the accuracy leverage to 99.66%.That is why there is a sceptical peak in the accuracy chart and a sudden decline in the loss plot. The FLOPs got for this model is 0.21G.

4.6. Fail cases

Although the best model gains a positive outcome, the recheck should be looked into to briefly review what the model missed or where the mistake could be. In this section, the matching is checked with top-1 accuracy, which found about 200 miss matched ones. As shown in Fig. 5 - the sample of fail cases suggests the model might be got complicated with the high contrast, high saturation image where the object is blending into the background. In other cases, multiple objects confused the model to identify which one to classify. Sub-species that share some akin should be tricky as well.

5. Limitation and Discussion

The hyper-parameters values are only applicable for this specific study. It can be reused but might need extra tuning to fit the desired result. The unchosen models do not mean that they are poor in general. By using the proper setup and tuning, the model will get decent performance. VGG model has the highest FLOPs in this observation. However, in a trade-off, this model returns a good result. It is one of

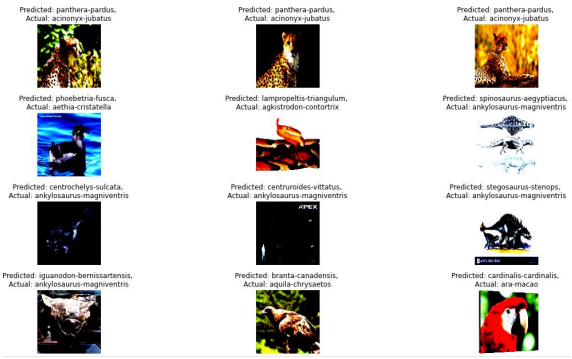


Figure 5. Some fail cases

the most popular pretrained models for image classification, which swiftly beat AlexNet after being introduced to the public. In the ICLR 2015 paper, VGG is introduced to work well with large-scale image and feature extraction [7].

ResNet and EfficientNet family are others worth trying models on different datasets. Apart from used models, they have many variants tailored for different purposes. This experiment does not run all of them to have a deeper comparison. Some models reach high accuracy before tuning, which opens the opportunities for others to achieve similar performance. In these particular tests, the accuracy is checked on the top-3 matching. The result between top-1 and top-3 will have a noticeable difference. More tricks and transformations should be applied to gain similar high accuracy like top-3 when using the top-1 matching. In training the best model, instead of running manually for freeze layers and unfreeze them, or further work like changing learning rate, this monitor could be automated scale to achieve.

Although models have used some image augments with flipping and rotation. However, other augmentation methods could be applied to gain absolute accuracy on top-3 matching or boost the outcome if top-1 matching is used. Some possible methods are adding noise, cropping, scaling, translation, brightness transforming, contrast transforming, saturation transforming and colour augmentation [6].

6. Conclusion

This experiment aimed to try various models in different structures, application of transformations, and hyperparameter tuning to achieve the possibly highest performance. The best-selected model got 99.66% for accuracy and 0.21 GFLOPs with the top-3 matching. Through these tests, it could be said that EfficientNet can provide a similar outcome to VGG16 and ResNet50 with fewer computing resources. There is some trade-off in the number of parameters between models. However, the accuracy can be boosted with the augmentations.

References

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 1

[2] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014. 2

[3] Vincent Lafage. Revisiting" what every computer scientist should know about floating-point arithmetic". *arXiv preprint arXiv:2012.02492*, 2020. 1

[4] Weibo Liu, Zidong Wang, Xiaohui Liu, Nianyin Zeng, Yurong Liu, and Fuad E. Alsaadi. A survey of deep neural network architectures and their applications. *Neurocomputing*, 234:11–26, 2017. 1

[5] Shagun Sharma and Kalpna Guleria. Deep learning models for image classification: Comparison and applications. In *2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, pages 1733–1738, 2022. 1

[6] Connor Shorten and Taghi M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):1–48, 2019. 5

[7] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2014. 1, 5

[8] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision, 2015. 1

[9] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *CoRR*, abs/1905.11946, 2019. 1, 3

[10] Maxime Vidal, Nathan Wolf, Beth Rosenberg, Bradley P. Harris, and Alexander Mathis. Perspectives on individual animal identification from biology and computer vision. *CoRR*, abs/2103.00560, 2021. 1