CV703 Assignment 1 - What kind of food is a bird dog?

Jose Renato Restom   Sara Pieri
Mohamed Bin Zayed University of Artificial Intelligence, UAE
{jose.viera, sara.pieri}@mbzuai.ac.ae

1. Introduction

In this assignment the goal was to design, train, and evaluate an image classification method on CUB [11], CUB + Stanford Dogs [5] and FoodX [4] datasets. We conducted several experiments with *traditional* ConvNets (AlexNet, SqueezeNet, ResNet, DenseNet), MLP-Mixer, ConvNext and Swin Transformer.

2. Literature Review

2.1. AlexNet

AlexNet is one of the traditional CNN architectures introduced by Yoshua Bengio in [6]. This architecture is mostly used nowadays as a way to benchmark the new developments since it has been outperformed by new models. The original LeNet-5 consisted of seven layers of convolutions arranged in a pyramidal manner (the deeper you go, the smaller the representation size).

2.2. SqueezeNet

SqueezeNet was created as a more efficient alternative to Alexnet. The authors created a CNN capable of obtaining comparable (sometimes even superior) accuracy with several times less parameters than the work proposed by [6]. The main advantage of using this architecture is the great trade-off between accuracy and model size, which allows SqueezeNet to have a *more efficient distributed training, and makes the deployment in embedded devices more feasible* [3].

The novelty of the architecture is the **Fire Module**, which consist of many 1x1 convolutional filters (squeeze blocks), followed by 3x3 filters (expansion blocks). The Fire module is illustrated in the Figure 1

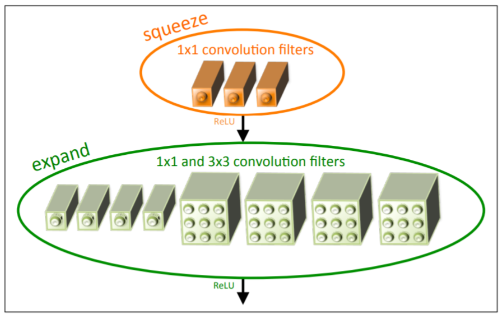

Figure 1. Fire Module. Image taken from 1

2.3. ResNet

The ResNet family of architectures have been in the panorama for a while. First introduced in the paper [2], the core idea behind these networks is the use of a *Residual leaning block* that utilizes a *Skip connection* in between layers. The Skip connection allows the network to deal with vanishing gradients while taking advantage of the identity function to improve performance without any significant increment in computational cost. The Residual block is shown in the Figure 2.

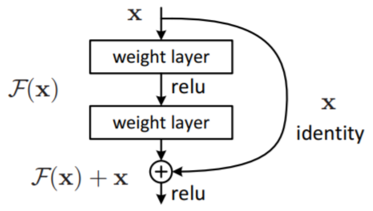
x
weight layer
relu
weight layer
$\mathcal{F}(\mathbf{x})$
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 2. Residual block with skip connection. Image taken from [2]

2.4. DenseNet

Building upon the idea of skip connections DenseNet was conceived using multiple connections from each layer to all the following ones. The authors stated that *DenseNets have several compelling advantages: they alleviate the vanishing-gradient problem, strengthen feature propagation, encourage feature reuse, and substantially reduce the number of parameters.*.

2.5. MLP-mixer

For years Convolutional models have been the default solution for computer vision tasks, and more recently the use of self-attention mechanisms in transformers have become a promising alternative. In comparison, not much research has been done with other options and that is precisely what the authors of [10] decided to attempt. The architecture they came up with, called *MLP-mixer*, does not require self-attention mechanism nor uses convolutions. Instead, the network learns by using multi-layer perceptrons on individuals patches (location features) and then using another MLP-layer acroos channels (spatial information).

2.6. ConvNext

Recently Vision Transformers represent state of the art in image classification task, being more accurate, efficient, and scalable than ConvNets. In [8] the authors have tried to gradually modernize a ResNet50 towards the design of a Visual Transformer (Swin Transformer [7]) to demonstrate that comparable performance can be achieved having similar training techniques and number of flops. Their design choices can be summarized in:

- **Macro design** Following the example of Swin Transformer, they adjusted the number of blocks to obtain a stage compute ratio of 1:1:3:1 and modified the stem cell to a 4×4 stride 4 convolutional layer.
- **ResNeXtify** Following ResNeXt's example [12], they introduced depthwise convolution in 3×3 conv layer in the bottleneck to increase the network width to the same number of channels as SwinT while maintaining a similar model complexity.
- **Inverted Bottleneck** They introduced an inverted bottleneck (see Fig. 3**b**). This is motivated by the fact that the Transformer's MLP block is four times wider than the input dimension and thus is comparable to an inverted bottleneck with an expansion rate of 4 as used in ConvNet architectures. [9].
- **Kernel Size** They increased the window size from 3×3 to 7×7 by moving the depthwise conv level up into the bottleneck block (see Fig. 3**c**). In this way the large conv kernel has fewer channels as the complexity falls on 1×1 conv. This approach is motivated by the fact that transformers are characterized by non-local self-attention which increases the size of the receptive field and that Swint T has reintroduced a window size of 7×7.
- **Micro design** They introduced several modifications layer-wise to match the design of Swin Transformer. Modifications include: replacing ReLU with GeLU, eliminating all GeLU layers in residual blocks except for the one between 1×1 layers, replacing BatchNorm with Layer Normalization, saparate downsampling by adding 2×2 layers with stride 2 between stages.

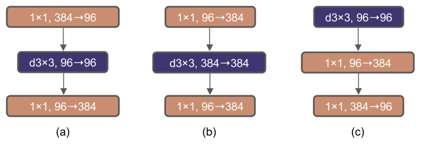

Figure 3. **Block modifications. a)** ConvNext block, **b)** inverted bottleneck, **c)** inverted bottleneck with moved up depthwise conv layer.

2.7. Swin Transformer

Introduced as part of the NLP, Transformers bring several challenges in image application. Among these is that the computational complexity of self-attention is quadratic to the image size. To obtain a linear complexity, Swin Transformer [7] was recently introduced. It splits the image into basic patches, gradually merging them in the following layers, and calculates self-attention locally within non-overlapping windows. Furthermore, it shifts the window partitioning between consecutive self-attention layers, resulting in new windows incorporating information about their spatial connection. An overview is show in Fig. 4.

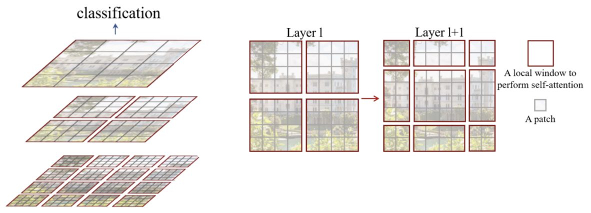

Figure 4. **Swift Transformer.** On the right, the hierarchical feature map obtained by merging patches. On the left, form level 1, $l + 1$ creates a new partition of the window providing information about connections.

3. Results

3.1. CNNs and MLP-mixer

We performed a manifold of experiments on all three datasets using AlexNet, SqueezeNet, DenseNet, Resnet and MLP-mixer architectures. All networks were pretrained on ImageNet-1K and subsequently finetunned on the corresponding dataset of interest. The results are shown in Table 3.1.

On AlexNet vs SqueezeNet One of the first interesting findings supported by the results is how on some of the tasks SqueezeNet was able to outperform AlexNet even though the latter possesses less than a third of number of the parameters of the first one. This shows how building optimized architectures from the ground can help not only get smaller networks, but also in some cases outperform larger and deeper networks.

model	T1	T2	T3
ResNet	**71.40**	56.53	63.64
DenseNet	68.97	55.38	**65.06**
MLP-Mixer	60.53	**68.72**	60.86
SqueezeNet	58.23	26.71	55.06
AlexNet	49.43	53.78	47.51

Table 1. **Accuracy 1 results for ConvNet models.**

Convolutions vs MLP As mentioned in [10], it seems that just using MLPs over the patches is a viable way to tackle computer vision tasks. What's more, the architecture is even able to reach comparable results to those of Resnet, Dense and even outperform Swin Transformer on the second task. These results support the idea of using other types of architectures that are not based on convolutions and self-attention mechanisms.

3.2. ConvNext

We conducted several experiments on ConvNext by finetuning the pretrained networks on ImageNet-1K.

ConvNext offers several variations (see Tab.3.2) which differ in the number of channels and blocks per stage.

model	FLOPs	#param	top-1 acc
ConvNext-T	4.5G	29M	82.1
ConvNext-S	8.7G	50M	83.1
ConvNext-B	15.4G	89M	83.8
ConvNext-L	34.4G	198M	84.3
Swin-B	15.4G	88M	83.5

Table 2. **Comparison between ConvNext and Swin Transformer.** Number of flops, parameters and top1 accuracy on ImageNet 1K are reported.

In the default setting, we finetuned ConvNext for 10 epochs with a learning rate of $4e-3$, batch size of 64, AdamW optimizer with decay 0.05, momentum 0.9.

Data augmentation Since the supplied datasets contain relatively few images, we have used the data augmentation provided in the official ConvNext repository. These include: resize, normalization, color jitter, center crop, RandAugment [1] and Random Erasing [15].

Data augmentation results to be a key as this resulted in a significant increase in accuracy. Also, it introduced greater stability in training. We noticed that without implementing data augmentation techniques, the parameter setting (e.g. learning rate) has a stronger impact in the training evolution and results. Therefore, all experiments on ConvNext will use this data augmentation setting from now on.

Models We tested the different configurations of ConvNext. The results obtained in terms of accuracy are close enough for ConvNext tiny small and base, while ConvNext large dramatically underfits the data. Results for the CUB + Dogs dataset are shown in Fig 5. The highest accuracy across different dataset was obtained with ConvNext-B. From now on we consider this as a reference model.

task	acc-1	acc-5	loss
T1 CUB	88.58	98.02	0.5
Dogs	93.08	99.59	0.3
T2 CUB + Dogs	90.59	98.9	0.4
T3 FoodX	70.63	88.26	1.5

Table 3. **Best Model results for ConvNext-B on all datasets.**

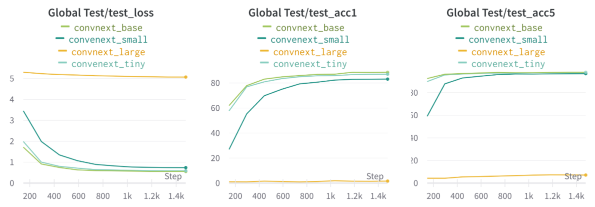

Figure 5. **ConvNext comparison on CUB.** Comparison on loss, top 1 accuracy and top 5 accuracy across ConvNext variants.

Learning Rate We changed the learning rate value to analyze the learning curves. Results for Cub + Dogs are shown in Fig. 6.

Regularization Techniques We have experimented with two different regularization techniques, MixUp [14] and CutMix [13], with different parameters. However the training did not benefit from their introduction accordingly with what mentioned in [8].

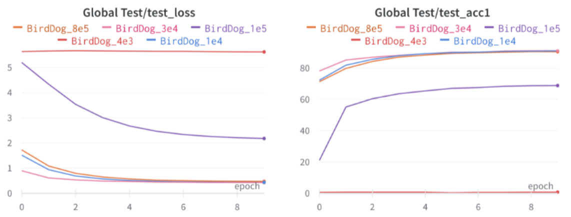

Figure 6. **Learning rate experiments on the Bird and Dog dataset using ConvNext-B.** Best results are obtained with a learning rate close to 0.0004.

3.3. Swin Transformer

We trained Swin Transformer on the FoodX dataset with a 64 batch size for 300 epochs. The model achieved an acc-1 63 and acc-5 88.55. The training lasted 18 hours. Not much can be deduced from this experiment since excessive training time did not allow us to conduct further tests (e.g. batch size).

References

[1] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmenta-

tion with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 702–703, 2020. 3

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1

[3] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016. 1

[4] Parneet Kaur, Karan Sikka, Weijun Wang, Serge Belongie, and Ajay Divakaran. Foodx-251: a dataset for fine-grained food classification. *arXiv preprint arXiv:1907.06167*, 2019. 1

[5] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Fei-Fei Li. Novel dataset for fine-grained image categorization: Stanford dogs. In *Proc. CVPR Workshop on Fine-Grained Visual Categorization (FGVC)*, volume 2. Citeseer, 2011. 1

[6] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 1

[7] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021. 2

[8] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. *arXiv preprint arXiv:2201.03545*, 2022. 2, 3

[9] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 2

[10] Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. *Advances in Neural Information Processing Systems*, 34, 2021. 2, 3

[11] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 1

[12] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017. 2

[13] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6023–6032, 2019. 3

[14] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017. 3

[15] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 13001–13008, 2020. 3