Assignment 3 : Kaggle competition

1. Introduction

This report summarizes my approaches for the Kaggle Competition during the MVA Course of Object Recognition, held in 2021.

2. Different Approaches

During this assignment, I wanted to try and apply some of the architectures we saw during class.

2.1. Convolutionnal Neural Network

First, I started with only a small CNN, using the model that was given and trying to fine-tune it, with different architectures, mainly consisting of a few convolutionnal layers at first, and then a classifier.
As expected, the results weren't great and I only obtained an accuracy of approximately 35%.

2.2. Transfer Learning

With these results, I understood that it would not be possible to fully train a network using only the 1000 images in the dataset. So the solution was to use a pretrained network, in order to extract meaningful patterns in the image, and only train a classifier on top of the network with the dataset.

2.2.1 AlexNet

I decided to first use AlexNet as a pretrained network. Indeed, it is considered as the first neural network, and I wanted to see how much I could improve the results obtained with this network by "upgrading" to more recent ones.
And, unsurprisingly, the results weren't that great either : I obtained around 50% of accuracy.

2.2.2 ResNet [1]

I then decided to use ResNet, which I already knew about from before the course, and used a small classifier on top of it (only 2 layers).
After some pre-processing (detailed in part 3), I was able to obtain over 90% of accuracy, which was my final submission.

2.2.3 VggNet [2]

Finally, I decided to use VggNet : even though the models aren't the most recent today, they seemed to be quite good and adapted to the task. In that case also, I trained a classifier on top of the existing network, but I also retrained the last layers to better fit the problem.
And with that network, I was able to attain some very good results (a little over 70% without any image pre-processing, and almost 90% with pre-processing).

3. Image pre-processing

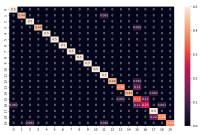
In order to improve the results, image pre-processing is required in order for the network to be more resilient to some changes in the input.
Here is the list of the steps the images go through:

- First, I resize the images to be 224×224 (the same input size as in the paper;
- Then, I randomly flip the image horizontally (with a probability of 0.5);
- After that, I randomly flip them vertically with a probability of 0.2, because only a few images of the birds are upside down;
- Then, I pad the images on 5 pixels, so that the border of some images (circular ones, for example) doesn't seem "new" to the network;
- Then I apply a random perspective to the images;
- And lastly, I randomly change the brightness, the contrast and the saturation of the images.

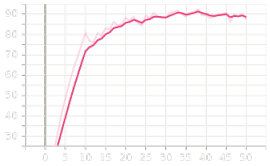
After all that, the images are normalized before going into the network.

4. Results

In the end, I obtained up to 91% of accuracy on the validation set. However, when I published the results on the Kaggle, it only reached up to 60%.
Below are the confusion matrix (which is clearly diagonal) and training error/validation accuracy of the network, using 65 epochs.



(a) Confusion Matrix



(b) Validation accuracy

References

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. 1

[2] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 1