AC50001: Convolutional Neural Networks in Keras

These exercises are from the Jupyter notebooks that accompany Chapter 5 of Chollet's book [1]. We have edited them so that they work on Colab with the latest version of Keras.

1. Setup

You should access the three notebooks at the links below and copy them to your own account.

5.1 - https://colab.research.google.com/drive/1K1QHBOsnCujGd0__QkCaJc4xiBnpY6Vj?usp=sharing

5.2 - https://colab.research.google.com/drive/18ONKytVnSp0kYzZAh2bMo7yJtOd3zv5V?usp=sharing

5.3 - https://colab.research.google.com/drive/1C4tf4TKfYUkGEvazxnyuzy-CQZ4nHv63?usp=sharing

You will then be able to run and edit them using your own profile. Make sure you select *GPU* under *Change Runtime Type* on the *Runtime* tab; this will enable you train convolutional networks considerably faster than on a CPU.

2. MNIST Digits

This notebook (5.1) trains and tests a convolutional neural network (CNN) on the task of classifying the MNIST digits. The methods used should look familiar to you. Follow the notebook through, running it, and making sure you understand the gist of it.

3. Dogs vs. Cats

This notebook (5.2) trains and tests CNNs on the task of classifying colour images as being of either cats or dogs. The data are from a Kaggle competition of several years ago (https://www.kaggle.com/c/dogs-vs-cats/overview).

One of the techniques it uses to help with overfitting is called *Dropout*. Dropout randomly selects neurons and sets their outputs to zero during training. This has been shown to reduce overfitting in a wide variety of problems. You can read about it in Section 4.4.3 of Chollet [1] or in the paper that proposed it [2].

For speed, this notebook only uses 10% of the original dataset. It will probably take about 5 minutes to train for 30 epochs. Training for 100 epochs with augmentation and dropout could take around half an hour. You might want to train for fewer epochs initially.

4. Pre-training

This notebook (5.3) explores the technique of using a convolutional network pre-trained on a larger, related dataset. The target task is again *Dogs vs. Cats*. The pre-training was done on a large image dataset called ImageNet (http://www.image-net.org/). It was done using a well-known CNN design called VGG-16 [3], the structure of which should look familiar.

Firstly, the notebook uses VGG-16 as a feature extractor, replacing its final dense layers with new ones and then training those layers on *Dogs vs. Cats*. This is fast.

Secondly, the notebook tries fine-tuning the whole network, making use of data augmentation. This will take much longer to run. So, while you are waiting…

5. Return to MNIST

The first notebook you tried in this session might have got a test accuracy of about 99.2% on the MNIST test dataset. That equates to classifying approximately 80 of the test images incorrectly. Can you do better by modifying the network or its training?

References

[1] Francois Chollet. Deep Learning with Python. Manning, 2018.

[2] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research 15(56):1929−1958, 2014. https://jmlr.org/papers/v15/srivastava14a.html

[3] Karen Simonyan, Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. International Conference on Learning Representations, 2015.