INFO371 Problem Set 7: Image regognition

March 7, 2022

Introduction, Data

This problem set is demanding and maybe less well defined. You are asked to do image recognition tasks, in particular detect the language of written text. The aim with this problem set is let you to play with convolutional networks, and understand the effect of the networks structure to the model capabilities and training time.

The training data consists of excerpts of printed text, 4100 images of Danish, 8500 images of English, 6500 images of Russian, and 5300 images of Chinese text. I try to add Thai text too, this is still work in progress. Your task is to develop neural network that is as good as you can to distinguishing between these languages. On top of these files, there are four times fewer testing images. The images can be downloaded from Google drive https://drive.google.com/drive/folders/1-p1hW3MYKfzT1uRki-ByzN6YfBLpEaq9?usp=sharing, The files are split into two (zipped) folders: *train* and *test*. Besides of the number of files in these folders, there is no real differences between training and testing images. All-in-all there is approximately 1.2GB of data.

The files are named as *source_LANG-xxx.jpg*. *source* refers to the source text (e.g. author and book name), *LANG* is language (*DA*: Danish, *EN*: English, *RU*: Russian, *TH*: Thai, *ZN*: Chinese), and *xxx* refers to alphabetically ordered chunk id. It is is easy to distinguish between English and Chinese, but much harder to separate English and Danish. The images are of different size, but 500×200 is a fairly common measure. All images are grayscale only, and all text is printed in the same point size (19 pt I believe).

Note also that this PS is groupwork. It is also more of a project than a problem set, many questions do not have a good answer, and there may be very different ways to achieve some of the results, and you may choose to do things in a different order and skip some of the tasks altogether.

1 Your Tasks

1. Download data and organize it in your computer. You may be able to use the UW server, but it only has 4G memory. If you have someone with a beefy desktop in the group then I'd recommend to use that one for this project, otherwise just pick the most powerful computer you have.

2. Browse the images of different languages so you will get an idea what we are working with.

Next, we get into more serious tasks. Here are the suggested tasks in the suggested order but you may do it differently, and you may skip some steps alltogether, or implement those in a completely different way if you have other good ideas.

1.1 Get the code to run

The first task is to get the code to run. We recommend to use a small sample of images (perhaps 1000), instead of the full set. Do not care about the performance, just get the code to work.

1. Write a neural network model that *runs* on the image data and outputs confusion matrix and accuracy. Let us not bother with good accuracy here, just get it to run. Feel free to adapt the the squares-circles, or cats-dogs example code (note: the latter uses multicolor images!).

2. At the end of your code, add predictions on validation data, print the confusion matrix, and compute the final testing accuracy. This is the final performance measure of your model.

1.2 EN-vs-ZN

Now it is time to attempt to improve the performance. Let's start slow and only distinguish between EN and ZN.

1. Play with your model parameters (number of layers, number of filters, kernel size, strides, pool size, number of nodes...), try different number of epochs. You can also try different image size, smaller images run faster but may lose too much information.

   Your task is to get the model to work reasonably well, reasonably fast, and distinguish the languages well enough.

2. Add a piece of code that plots a few misclassified images. Can you see why are these images misclassified?

   See the squares-circles code for an example.

3. Comment your results. What did you try? How good were the best results? What kind of problems did you run into?

1.3 Crop images

The images in the dataset are of different size. In the cats-dogs example, we resized those into the same dimension. You can do this here as well, but it may be better to *crop* the images into the same dimension. This is because in case of cats and dogs, we want to preserve the whole image to determine which animal is there. However, in case of the languages, only a few letters may be sufficient to distinguish between EN, RU, and ZN (DA may be more tricky).

1. Write code to crop that reads both the training and testing images, crops those, and writes the cropped images back to disk in a different folder. You may choose something like 100×100 or 200×200 size.

   We recommend to use PIL library, see e.g.
   https://www.geeksforgeeks.org/python-pil-image-crop-method/. But you can use something else too, e.g. I am using *convert* from image magick in a shell script.

1.4 Other languages

When you are done with EN-ZN, start adding other languages. Can you distinguish between EN, RU and ZN? Can you do it when adding DA to the mix?

- As above, explain what did you try, what were your best results, and what kind of problems did you run into.