Deep Learning — Assignment 13

Assignment for week 13 of the 2023 Deep Learning course (NWI-IMC070) of the Radboud University.

Names: Luka Mucko, Luca Poli

Group: 46

Instructions:

- Fill in your names and the name of your group.
- Answer the questions and complete the code where necessary.
- Keep your answers brief, one or two sentences is usually enough.
- Re-run the whole notebook before you submit your work.
- Save the notebook as a PDF and submit that in Brightspace together with the `.ipynb` notebook file.
- The easiest way to make a PDF of your notebook is via File > Print Preview and then use your browser's print option to print to PDF.

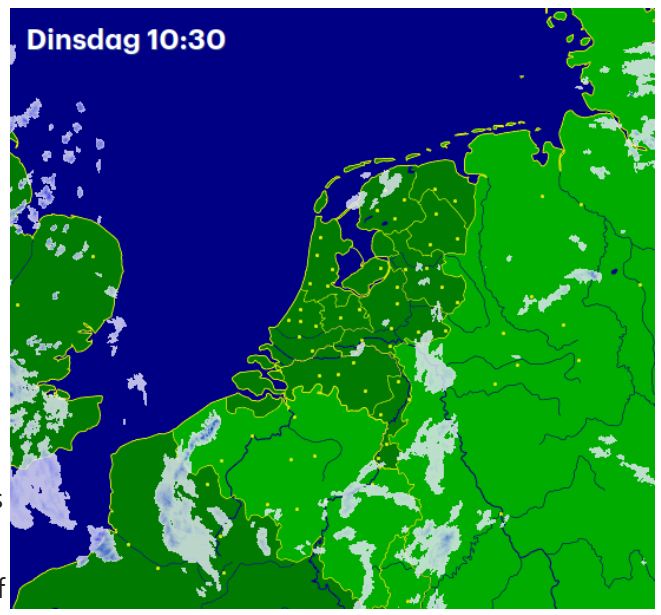
Objectives

In this assignment you will

1. Think about the design of deep neural networks on a higher level
2. Read papers that use deep learning in an application

This week's assignment does not involve any programming.

13.1 Predicting the weather (7 points)



Most people don't like to be oustide in the rain. Fortunately, we nowadays have radar sensors, that can show the amount of precipitation over a wide area in real time. Weather services like buienradar use this data to forecast when it will rain. But the models they use are often quite simple: just move the rainclouds according to the current wind patterns.

In this assignment you will investigate whether you you can do better with deep learning.

More concretely: The task that we have in mind is to predict the amount of rain up to 3 hours in advance, for the entire country of the Netherlands. The input will be radar images of the amount of rain *currently* falling. For training, there is historical radar data going back several years. You may assume that the data has been gathered with a sample rate of 5 minutes, and at a resolution of 1500m per pixel. See the KNMI website for more information.

(a) Are there risks of unfair biases in this task? Explain your answer. **(1 point)**

Yes, there are risks of unfair biases. For example, the model would be biased towards the usual weather patterns in the Netherlands; so, if the weather patterns is quite different at the time of prediction (given the climate change, its likely), the model would not be able to predict the weather correctly. Other biases could be related to the measurements and the device used for that. We could also add more information like the temperature, the humidity, type of terrain, time of day/month/year, ecc.

(b) Suppose we want to use a deep neural network for this task. How would the input data be represented? **(1 point)**

For now, consider using only the current radar data as input.

The input data would be the radar image, represented as a matrix of pixels, where each pixel is a number representing the amount of rain in that area.

(c) What are the targets for prediction? How are they represented? **(1 point)**

The target is the amount of rain in the next 5 minutes, represented as a matrix of pixels, where each pixel is a number representing the amount of rain in that area. Then we will use this result as input for the next prediction up to 3 hours in advance. Otherwise, we could predict the rain in 3 hours in advance directly.

(d) If we want to make a prediction for the entire country of the Netherlands, how large should the input be to contain all the relevant information? **(1 point)**

The radar image would be represented by a matrix of at least 18600 pixels (where each pixel contain the amount of rain), because we should add also some area around the Netherlands to have a complete prediction.

Rainclouds act differently depending on the terrain. So it can be useful for the model to know which areas of the radar image are above sea, rivers, forests, cities, mountains (as if), etc.

(e) How can you include this information in the input to the model? (1 point)

If we want also to include the type of terrain information, we could add another matrix of 18600 pixels, where each pixel is a number representing the type of terrain in that area; so, after some convolutional layers, we could concatenate the two matrices and then continue with the other layers. Otherwise, we could make each pixel as a vector of the amount of rain and the type of terrain in that area (possibly coded with hot encoding).

(f) If information on the type of terrain is not available, could it be learned instead? If so, how? (1 point)

Yes, we could use an unsupervised approach (if we can only use this dataset). The main idea would be to use a clustering model that create cluster of pixel with similar weather behavior (given the amount of rain at one time and the amount of rain at the next time). Then, we could use the clusters as a representation of the type of terrain.

The weather also depends on the time of day, in particular on whether the sun is shining. And on the time of year, especially the temperature.

(g) How could the time of day and time of year be given as inputs to the network? (1 point)

We could represent these new inputs as a vector of 6 elements (day, month, year, hour, minute, temperature), then we could concatenate this vector with the image (as a new channel if we are using a CNN).

13.2 Weather prediction models (4 points)

Consider a model that uses only the current radar data and terrain as input, to predict rain 3 hours in the future.

(a) What kind of model would you use for this task? (1 point)

We would use a model that uses CNNs to predict the weather in the future given only current data and no previous data.

(b) How would the performance of a model that takes only a single radar image as input compare against the simple baseline model that moves the radar image in the wind direction? (1 point)

A model that uses only a single radar image as input would likely outperform a simple baseline model because the deep learning model can capture more patterns in the data other than just wind direction.

To improve the model, it makes sense to take the temporal aspect into account, by including historical data. So radar scans from 5 minutes ago, 10 minutes ago, etc.

(c) Give two ways to include this additional data into the model. (2 points)

1. Utilize recurrent layers such as RNNs or LSTMs
2. 3D convolutions use a 3rd dimension other than the 2d radar image which could be time.
3. ConvLSTM uses both CNNs for capturing spatial data and LSTM for capturing temporal data.

13.3 MetNet (4 points)

We are not the first people to think about this problem. The paper MetNet: A Neural Weather Model for Precipitation Forecasting has tried to tackle the task of predicting rain from radar images.

(a) Have a look at the MetNet paper and compare their method to your answers in 13.1 and 13.2. What are they doing the same, what are they doing differently? (3 points)

Input encoding:

- Like us (in 13.2b), they use a timeseries of radar images (not every 5 but every 15 minutes); however, they also provide other information such as: 16 spectral bands from GEOS-16, real values for longitude and latitude, elevation, and the time of the day, month and year (also the target time, to condition the model).

Output encoding:

- The output, unlike us, is the predicted weather at the conditioned time (which is an extra input); in this way they can train and use the model to predict, in one-shot, the weather at any time (up to 3 hours) in the future.

Model:

- Like we suggest in 13.2c, they use a CNN to extract features from the radar images and a LSTM to capture the temporal aspect of the data. However they use a more complex structure (with more layers and spatial downsample and aggregation) then us.

(b) Can the trained MetNet be used in the Netherlands using data from KNMI, without retraining? How? Or why not? (1 point)

The MetNet is trained on data from the US, so it would be difficult to use it in the Netherlands without retraining. Since they use latitude, longitude and elevation as input, it could be possible to use the model without retraining if we can find a way to map the data from KNMI to the data from the US.

13.4 MetNet - version 2 and 3 (4 points)

You may have noticed that the MetNet paper came out 3 year ago (you can tell the year from the `/YYMM.NNNNN` arxiv url). After that two new models have been released by the same group:

- MetNet-2: Skillful Twelve Hour Precipitation Forecasts using Large Context Neural Networks
- MetNet-3: Deep Learning for Day Forecasts from Sparse Observations

(a) What method is used by MetNet-2 to get a larger receptive field in the convolutions? (1 point)

MetNet-2 uses dilated receptive fields, whose size doubles layer after layer, in order to connect points in the input that are far apart one from the other source. The size of the dilation factor doubles each layer from 1,2,4,...,128.

(b) How many parameters do the original MetNet and MetNet-3 have? Is one model significantly larger than the other? (1 point)

MetNet has 225M parameters and MetNet-3 has 227M parameters. So they are of the same size, relatively.

(c) Estimate how much CO_2 was emitted for training MetNet-3. (2 points)

Hint: You can find information on the power consumption of google TPUs here (the reported numbers are per chip, not per pod).

Hint 2: Make a reasonable assumption for the CO_2 / kwH (see lectures, or here)

"The network is trained on 512 TPUv3 cores, where each of the 32 groups of 16 TPU cores process 2 input-output samples and the gradients from each group are synchronously aggregated after processing each batch. The fully trained MetNet-3 model took 7 days to train."

Using that data and this calculator We can see that for 512 TPUv3 we get 283W * 168h * 512 = 24340 kWh.

If we assume the model was trained in the USA (given all the examples are of the US) the estimated kgCO2 released goes from 5841kg to 13800kg (see tables at the end of this paper). If we assume the model was trained in the Netherlands using the data from the European union the Netherlands released 321g/kWh in 2022 giving us 7813kg of CO2 released.

The end

Well done! Please double check the instructions at the top before you submit your results.

This assignment has 19 points.

Version 6262555 / 2023-12-11