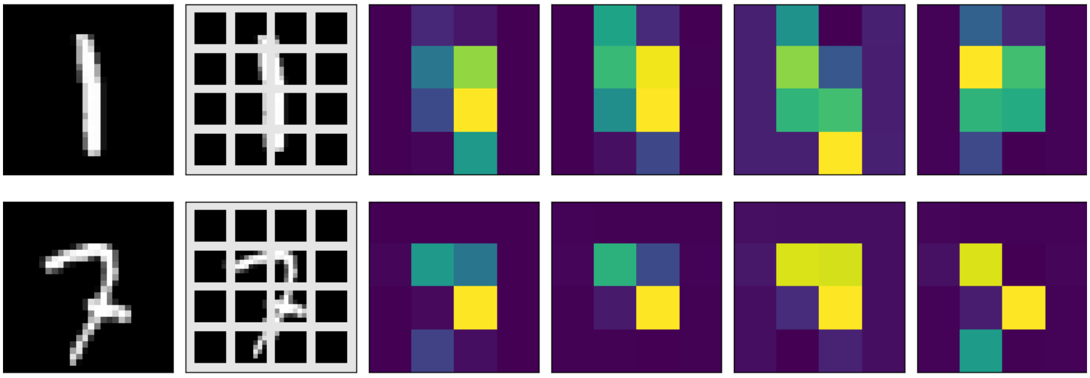
NNDL2024: Exercices & Assignments
Session 4 (14 Apr)

Mathematical exercises

1. (20 points) Vision Transformer.

   (a) Below is an example visualization of the attention mechanism for a ViT classifier trained on MNIST digits, but unfortunately the lecturer forgot to add all explanations. Think about how the model works and what the lecturer might want to show when explaining the model, and then proceed to **write a caption for the image**. Be specific, explaining in detail what exactly the plots might be showing using the right concepts and mathematical notation when applicable, and also explain how to interpret the plots. What can you say about the model itself based on these plots?



   (b) ViT partitions the image into a regular grid of sub-images or patches, splitting the 28×28 MNIST digit images into 16 patches of size 7×7, or perhaps 49 patches of size 4×4. Explain how the model would need to be changed if you split the image into horizontal rows instead, so that you would have 28 sub-images of size 1×28. Do you think the model would work roughly as well, or did we lose something by switching to image stripes instead of patches?

2. (20 points) Find one scientific article that uses transformers for modelling *some other type of data than language, images or videos.* Provide the title, author names and url. Read the paper and describe how they use it, with some pointers to the aspects to cover provided below. Do this properly yourself rather than asking LLM to summarize it for you even though it would here do a good job – the whole point of the exercise is to learn to read scientific papers on the topic!

   - What is the task? For instance, what do they try to predict.
   - How is the input data processed? Is is a set or a sequence? Explain key preprocessing steps needed to convert the data into a suitable format (e.g. tokenization in language models).
   - What kind of architecture was used? How many layers and parameters? Are there some interesting non-standard components?
   - How was the training done? Supervised? Self-supervised? Is some form of transfer learning used? How much data was used?
   - Did it work well? What was the method compared against and how did it fare?

- Your subjective feeling about the paper: Is the model good? Was the paper clear in communicating these aspects?

3. (20pts) Suppose we observe a bivariate Gaussian data $(x, y) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ with

$$\boldsymbol{\mu} = (0, 0), \boldsymbol{\Sigma} = \begin{pmatrix} 1 & c \\ c & 1 \end{pmatrix}$$

for some constant c. Now, we want to estimate c. Let's devise a very simple self-supervised algorithm: Learn a linear regression by least-squares to predict y from x. Show (mathematically, not empirically) that this learns the parameter c. Discuss how you can interpret this as self-supervised learning.

Computer Assignment

1. (40pts) Self-attention. Implement the multi-head self-attention operation, taking in a set of N vectors of D dimensions and outputting a matrix of the same size. Do this without relying on neural network libraries, but rather write directly the required operations in `NumPy`. Note that there are some differing conventions on which way the attention equations are written – complete this task using the 'QKV attention' as described in the notebook, for ease of grading.

   (a) Implement a function that performs the standard self-attention operation. Include the dot product scaling, but do not add any extra components (positional encodings etc).

   (b) Implement the multi-head version, so that you use the function you implemented above as a component. You can assume that the number of heads H is chosen so that D/H is an integer.

   (c) Run your code for the data (with $N = 5$ and $D = 6$) provided in the notebook with $H = 1$ and $H = 3$. For $H = 3$, use identity as the final transformation and use the same query/key/value matrices as for $H = 1$ but interpret them so that the first H columns correspond to the first head etc.

   (d) **Report:**

   - The *attention weights* and the *output* of the operation for $H = 1$ and $H = 3$.
   - What happens if you change the order of the first two inputs? Check by running the code and explain your finding.
   - For both $H = 1$ and $H = 3$ we use the same weight matrices, yet the result is not identical. Explain why this is, referring also to the mathematical expressions. Is one of the models more expressive?