Very Deep Learning **Technische Universität Kaiserslautern**
WS 2022/23 Dr. Muhammad Zeshan Afzal

Exercise 1 - Computational Graphs and Gradients

Deadline: 14.11.2022 **Total Marks: 30**

Submission

- All submissions will be handled through OLAT.
- Your submission should contain a PDF with the solutions to the exercise questions (and any Python code files) zipped together in a single file.
- Include the group number along with the names and matriculation numbers of all group members on the PDF.
- For Jupyter notebook, please save it with the outputs of your code displayed.
- The points obtained in the bonus exercise will only be added to your total if the total does not exceed the exercise total. You can see this as a way of making up for any points lost in the other sections.

1.1. Basics [1+1+1+1+1=5]

i) What is deep learning? What makes a neural network „deep”?

ii) What do you understand by the term backpropagation?

iii) Why do we use activation functions in neural networks?

iv) What is the difference between supervised and unsupervised learning?

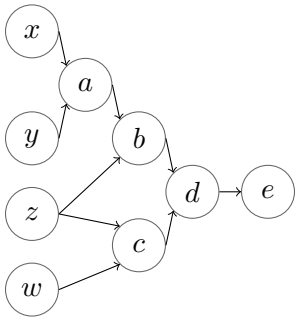
v) What is a Tensor in PyTorch and how is it different from a matrix?

1.2. Computational Graphs [3+3+2+4=12]

Computational graphs are directed graphs that represent the dependencies between the variables and operations within a model or, more generally, a mathematical expression. As an example, give the expression

$$f(w, x, y, z) = 4(xyz + max(z, w))$$

it has the following computational graph:



where $a = xy$, $b = az = xyz$, $c = max(z, w)$, $d = b + c$ and $e = 4d$.

Tasks: Now, consider the following expressions:

$$\mathcal{A} = (a - (b \cdot c)) \cdot d$$
$$\mathcal{B} = \sqrt{a + b + c^3} + \log(\frac{a}{d})$$
$$\mathcal{C} = d - \exp^{(a+b)\cdot c}$$

i) Introduce intermediate variables **for all three expressions** such that each variable represents a single mathematical operation.

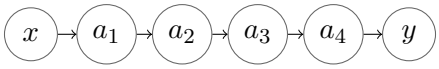
ii) Using your chosen intermediate variables, draw computational graphs **for all three expressions**.

iii) Given that $a = 2$, $b = -7$, $c = 4$ and $d = 1$. Perform forward propagation on **any two** of the three computational graphs you created in (ii).

iv) Perform backpropagation on the **same two** computational graphs you chose in (iii).

1.3. Vanishing and Exploding Gradients [1+3+2+1+3=10]

Consider a network with input $x \in \mathbb{R}$, 4 hidden layers each having only one node, and one output $y \in \mathbb{R}$:



In the network each node corresponds to the sigmoid of the preceding node multiplied with some weight: $a_i = \sigma(w_i \cdot a_{i-1})$, $i = 1, \ldots 5$, where a_0 corresponds to the input x and a_5 corresponds to the output y.

The sigmoid function is given by

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Tasks:

i) What is the vanishing gradients problem?

ii) Calculate the derivative σ' of the sigmoid function. What are the minimum and maximum values of the derivative?

iii) By using the chain rule, calculate the gradient $\frac{\partial y}{\partial x}$.

iv) Does the maximum value of the derivative of sigmoid contribute to the vanishing gradients problem? If so, how?

v) What is meant by exploding gradients? Why do we not want the gradients to explode? When can sigmoid activations have an exploding gradient?

1.4. Handwritten Digit Recognition Using PyTorch [3]

Follow the instructions in the Jupyter notebook `exercise_1.ipynb` to perform handwritten digit recognition using PyTorch. The notebook contains detailed instructions about downloading the dataset, creating the neural network, training it, and evaluating the results.

Your task is to fill in the missing code annotated with `TODO` tags in the comments. You can get up to 3 points if you can fill in all the missing code, successfully train the network and get an accuracy of around 90% on the test data.

As a bonus exercise, if you can do modifications to the network architecture or the training parameters such that you can get an accuracy above 95%, you can get 2 additional points.

Hint: You can import the notebook in Google Colab to get started with minimal setup or to use GPUs.