**Department of Artificial Intelligence**

**College of Computer and Cyber Sciences**

**Introduction to Deep Learning**

***Backpropagation with PyTorch***

1. **Learning Objectives**

By the end of this lab, students will:

* To be able to compute Backpropagation with PyTorch.
* To be able to compute Gradients with PyTorch.

1. **Lab Requirements**

Software:

* Python or Anaconda Jupyter notebook.
* PyTorch.
* CUDA.

Hardware:

* Students should use the lab devices for running their deep learning models.

1. **Explanation of Key Concepts**

* **Backpropagation & Gradients:**

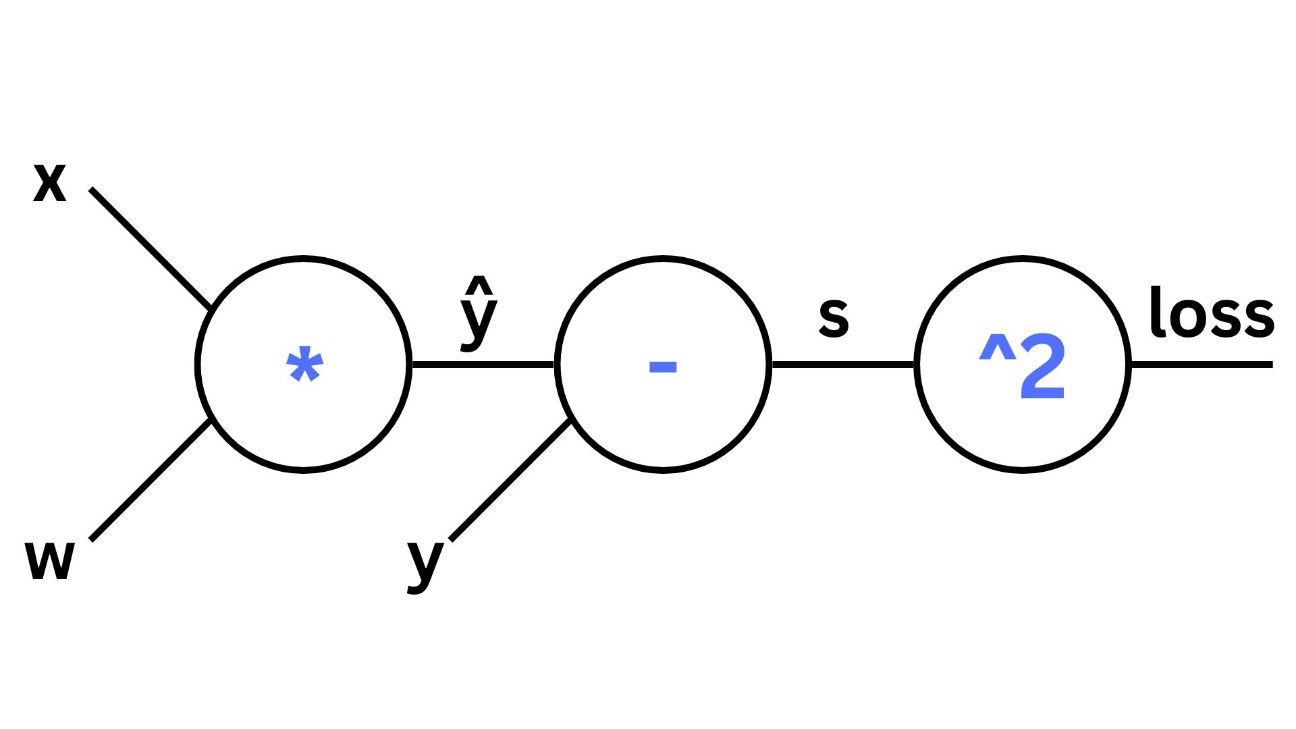
Backpropagation is an essential algorithm used to train neural networks by adjusting weights and biases. It works by calculating the error (loss) between the model's prediction and the actual value, then propagating this error backward through the network to compute the gradients of the loss with respect to each parameter. These gradients tell the model how to adjust the weights and biases to minimize the error.

Once the gradients are calculated, the model uses gradient descent to update the weights and biases by moving them in the direction that reduces the error. This process is repeated iteratively, improving the model's predictions over time by reducing the loss and refining the model's parameters.

1. **Activities**

* **Exercise 1: Computing the Gradient and Backpropagation.**

Using the network diagram provided, define the input, target, and weight tensors. Calculate the output and loss. Apply backpropagation to compute the gradient of the loss with respect to the weight.



|  |
| --- |
| # Code provided in the notebook |

Screenshot of the result:

A screenshot of a computer

AI-generated content may be incorrect.

1. **Tasks**

* **Task 1:**

In this task, you will modify the code from Exercise 1 to apply the following functions, and correctly choose which variables should have requires\_grad=True to compute gradients (Make sure to reset the gradients before each function):

1. Function 1:

2. Function 2:

3. Function 3:

Add your code here (or just submit your jupyter notebook):

|  |
| --- |
|  |

Screenshot of the result:

|  |
| --- |
|  |

1. **References**

[How to compute gradients in PyTorch?](https://www.tutorialspoint.com/how-to-compute-gradients-in-pytorch)

[PyTorch Tutorial 03 - Gradient Calculation With Autograd](https://www.youtube.com/watch?v=DbeIqrwb_dE&list=PLqnslRFeH2UrcDBWF5mfPGpqQDSta6VK4&index=4)

[PyTorch Tutorial 04 - Backpropagation - Theory With Example](https://www.youtube.com/watch?v=3Kb0QS6z7WA&list=PLqnslRFeH2UrcDBWF5mfPGpqQDSta6VK4&index=6)