1. **Write the Python code to implement ReLU.**

if a > 0: return a else: return 0

1. **What is the "hidden size" of a layer?**

Repræsenterer antallet af neuroner i hvert lag i de skjulte lag.

1. **What does the t method do in PyTorch?**

Transpose

1. **Why is matrix multiplication written in plain Python very slow?**

For loops for daaays

Pytorch bruger C++ frem for Python, da Python er et langsomt sprog.

1. **In matmul, why is ac==br?**

Der er to matricer. Førset matrices kolonner (output?) skal være samme størrelse som anden matrices rækker (Input?)

og hvorfor? måske fordi det ellers ikke virker?

1. **In Jupyter Notebook, how do you measure the time taken for a single cell to execute?**

%time

1. **What is "elementwise arithmetic"?**

basis operationer bliver lavet for hvert enkelt element hvis begge tensor har samme shape.

Virker kun med samem shape, eller når de er broadcastable

1. **Write the PyTorch code to test whether every element of a is greater than the corresponding element of b.**

(a > b).all()

1. **What is a rank-0 tensor? How do you convert it to a plain Python data type?**

sum() og mean() er en reducerings operationer, som retunerer tensors med kun et element, som kunne være en boolean eller et nummer. Hvis man vil konventere det til plain python, skal man kalde på funktionen .item

(a + b).mean().item()

1. **What does this return, and why? tensor([1,2]) + tensor([1])**

Det retunerer: tensor([2,3]) da de er broadcastable og dermed bliver denne tensor på højre side lavet større.

1. **What does this return, and why? tensor([1,2]) + tensor([1,2,3])**

RuntimeError, da de ikke har samme størrelse og ikke er broadcastable

1. **How does elementwise arithmetic help us speed up matmul?**

Vi kan slette et loop ved at gange noget med noget andet

I Stedet for at lave den sidste operation manuelt med loop nummer 3, hvor vi tager hvert element af de to tensor og ganger det ene med det andet, gør vi brug af elementwise arithmetic, som laver præcis det internt. Derfor kan der slettes et loop

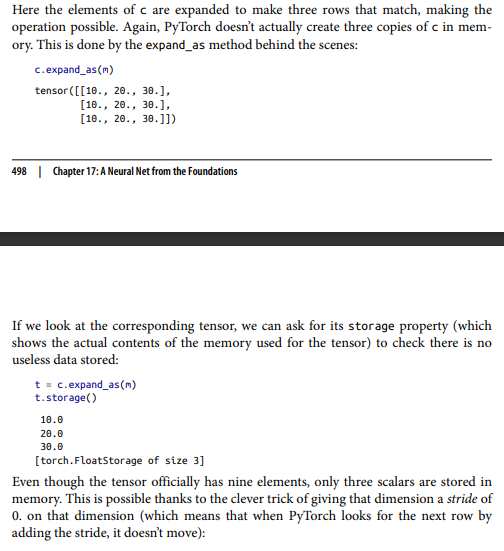
Black magic og matematik!

1. **What are the broadcasting rules?**

PyTorch sammenligner formen af elementer. Starter med at bruge *trailing dimensions* og arbejder baglæns. 2 dimensioner er kompatible når:

* 1. De er lige (ens)
  2. En af dem er 1, hvor i det tilfælde den dimension er broadcast til at være det samme som den anden.

1. **What is expand\_as? Show an example of how it can be used to match the results of broadcasting.**



1. **How does unsqueeze help us to solve certain broadcasting problems?**

Det giver muligheden for at broadcaste i en anden dimension, altså kolonner i stedet for rækker.

1. **How do we show the actual contents of the memory used for a tensor?**

tensor.storage()

1. **When adding a vector of size 3 to a matrix of size 3×3, are the elements of the vector added to each row or each column of the matrix? (Be sure to check your answer by running this code in a notebook.)**

Ret sikker på at det er på rows

1. **Do broadcasting and expand\_as result in increased memory use? Why or why not?**

Nej fordi at selve broadcastet ikke gemmes i en tensor i hukommelsen.

1. **What are the forward pass and backward pass of a neural network?**

Forward pass: Udregningen. Værdi af output fra input, udregner fra første neuron til sidste. Det er modellens gæt

Backward pass: Udregner forandring i vægtene using gradient descent. Backpropagation. Det er modellens forsøg i at blive klogere

1. **Why do we need to store some of the activations calculated for intermediate layers in the forward pass?**

Således at backward pass kan finde activation function

1. **What is the downside of having activations with a standard deviation too far away from 1?**

Activations ender enten med at blive nan eller 0

1. **How can weight initialization help avoid this problem?**

Ved at holde standard deviation af activations på 1, så man får brugbare tal.

1. **What is the formula to initialize weights such that we get a standard deviation of 1 for a plain linear layer, and for a linear layer followed by ReLU?**

Linear: 1/kvadrat rod(n\_in).

Med ReLU: 2/nin

1. **Why do we sometimes have to use the squeeze method in loss functions?**

For at fjerne dimensioner med en længde af 1. Altså er det en metode til at reshape en tensor.

1. **In what order do we need to call the \*\_grad functions in the backward pass? Why?**

mse, relu, lin

because chain rule

1. **What is \_\_call\_\_?**

Indbygget Python metode. Gør at man kan skrive klasser hvor instansen er en metode og kan blive kaldt som en metode

1. **If you are mathematically inclined, find out what the gradients of a linear layer are in mathematical notation. Map that to the implementation we saw in this chapter.**

Det er vi ikke.

1. **Implement everything in this chapter using NumPy instead of PyTorch…**

Boi, We have no time!