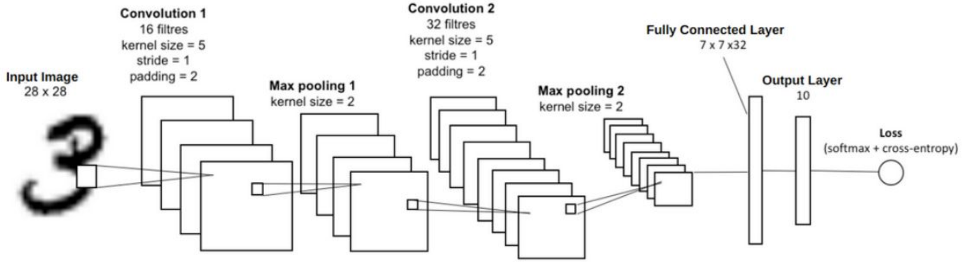
Quebec
Artificial
Intelligence
Institute
Mila

Tutorial: Week 3

Image Classification and Object Detection with CNNs

Jeremy Pinto
jeremy.pinto@mila.quebec

CNN Tutorial



How many trainable parameters do we have in total?

In this tutorial you will:

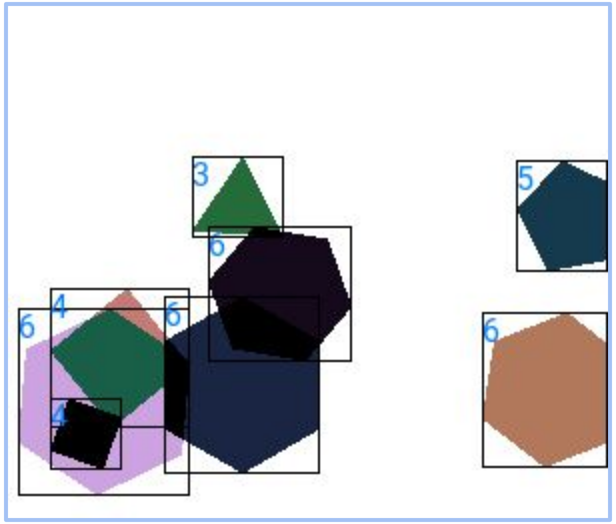
Part 1:

- Explore the MNIST digit dataset
- Implement and train an MLP to classify digits
- Implement and train a CNN (LeNet) to classify digits
- Use regularization on your models (batch norm)

CNN Tutorial

Part 2 (optional):

- Finetune your own object detector on a toy object detection dataset
  - Faster R-CNN for object detection
  - Mask R-CNN for semantic segmentation

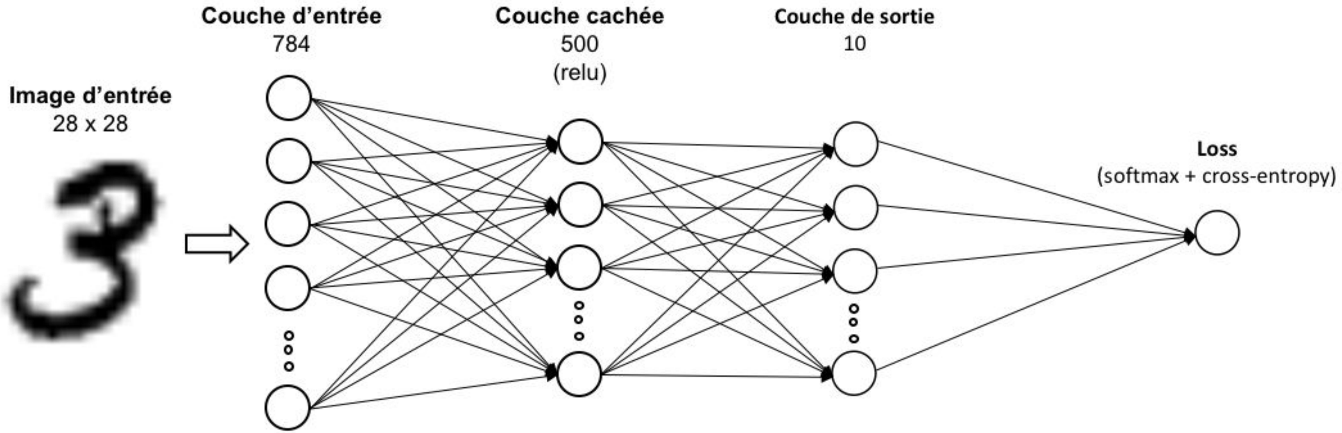


Quebec
Artificial
Intelligence
Institute

Mila

Tutorial Questions

Tutorial Questions



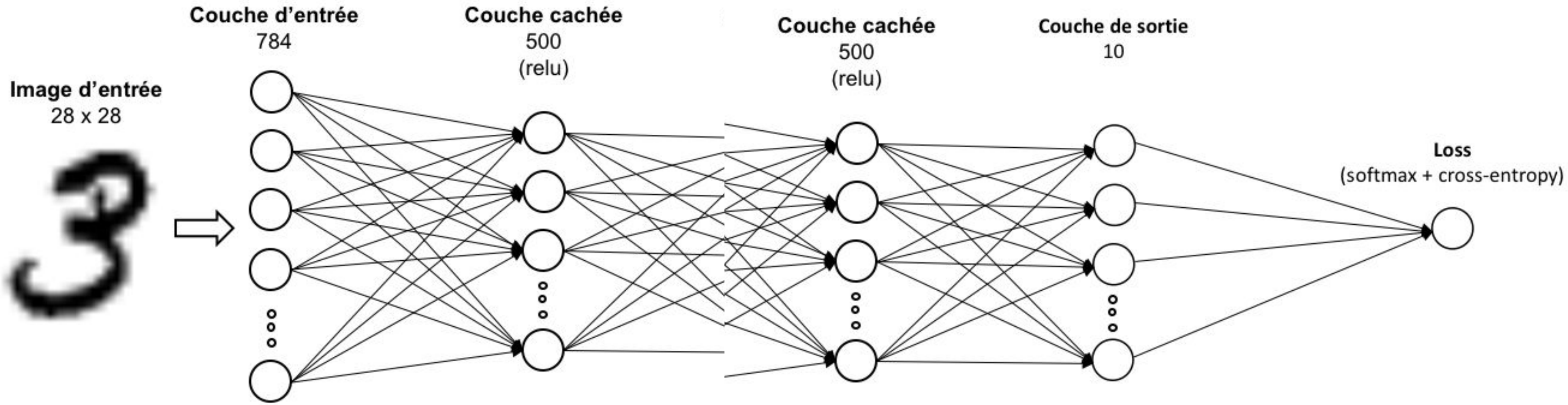
The figure is misleading - it actually has 2 hidden layers (refer to the code)

```
MLP(
  (hidden_layer): Sequential(
    (0): Linear(in_features=784, out_features=500, bias=True)
    (1): ReLU()
    (2): Linear(in_features=500, out_features=500, bias=True)
    (3): ReLU()
  )
  (output_layer): Sequential(
    (0): Linear(in_features=500, out_features=10, bias=True)
  )
)
```

```
Number of parameters in the model (weights and biases):  648010
```


Mila

Tutorial Questions

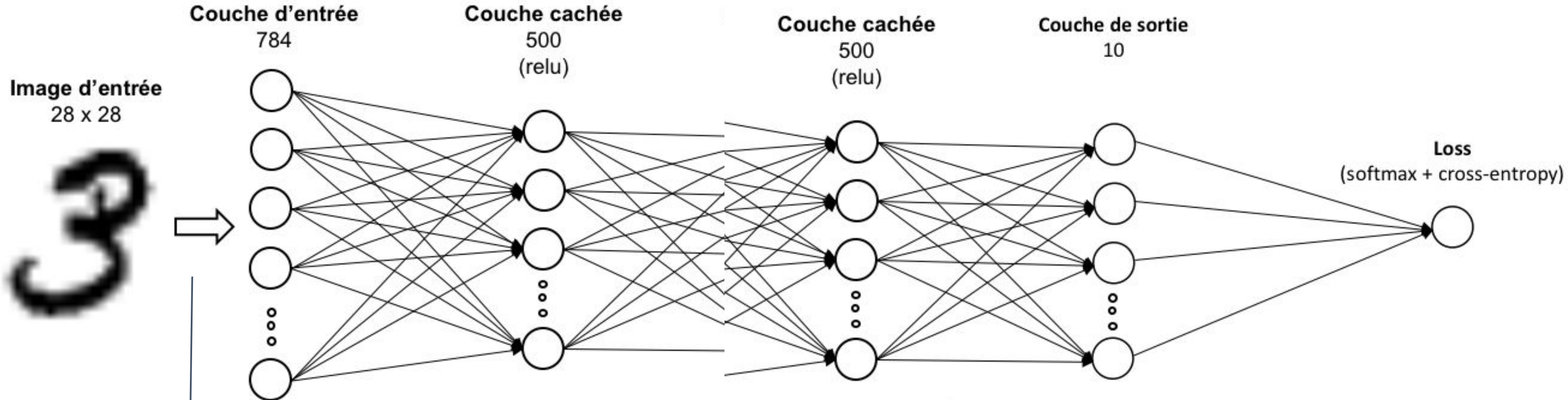


What is should look like:

```
MLP(
  (hidden_layer): Sequential(
    (0): Linear(in_features=784, out_features=500, bias=True)
    (1): ReLU()
    (2): Linear(in_features=500, out_features=500, bias=True)
    (3): ReLU()
  )
  (output_layer): Sequential(
    (0): Linear(in_features=500, out_features=10, bias=True)
  )
)
```

```
Number of parameters in the model (weights and biases):  648010
```

Tutorial Questions

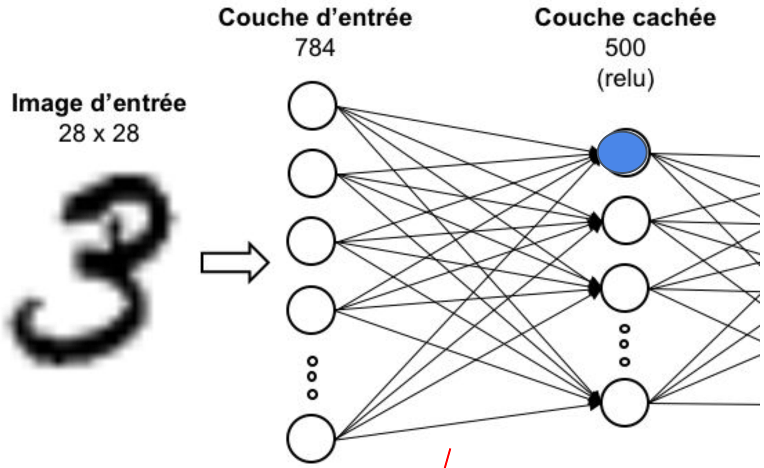


Reshape - not a layer!

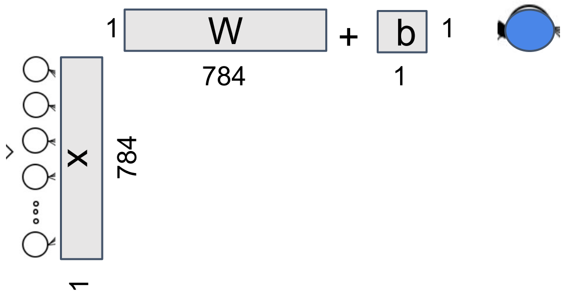
```
MLP(
  (hidden_layer): Sequential(
    (0): Linear(in_features=784, out_features=500, bias=True)
    (1): ReLU()
    (2): Linear(in_features=500, out_features=500, bias=True)
    (3): ReLU()
  )
  (output_layer): Sequential(
    (0): Linear(in_features=500, out_features=10, bias=True)
  )
)
```

```
Number of parameters in the model (weights and biases):  648010
```

Tutorial Questions



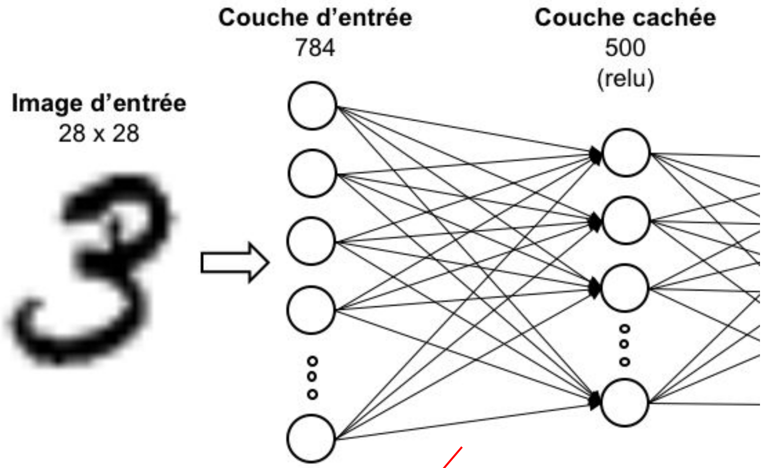
784 to 500 - 1st layer

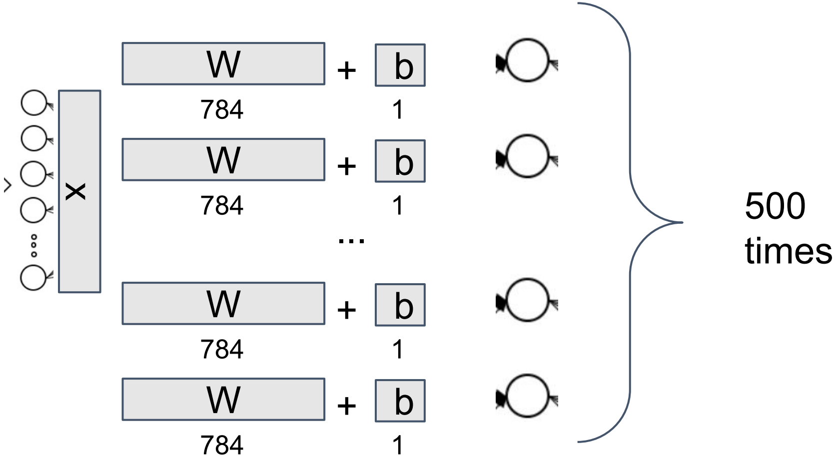



```
MLP(
  (hidden_layer): Sequential(
    (0): Linear(in_features=784, out_features=500, bias=True)
    (1): ReLU()
    (2): Linear(in_features=500, out_features=500, bias=True)
    (3): ReLU()
  )
  (output_layer): Sequential(
    (0): Linear(in_features=500, out_features=10, bias=True)
  )
)
```

```
Number of parameters in the model (weights and biases):  648010
```

Tutorial Questions





784 to 500 - 1st layer

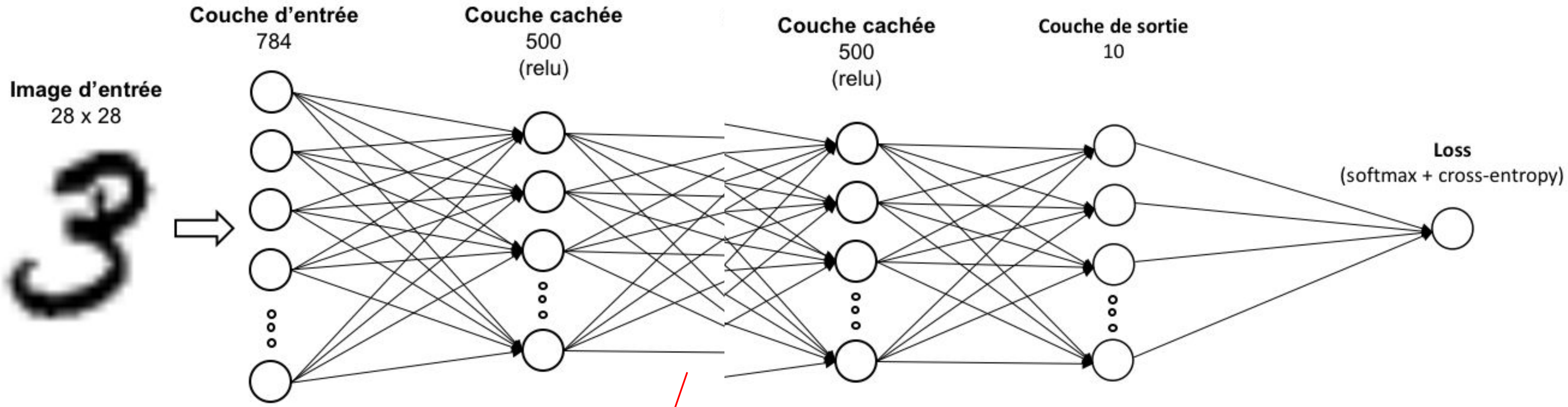
1st hidden layer:

500 vectors of 1x784
+ 500 biases
= 500*785
= 392 500

```
MLP(
  (hidden_layer): Sequential(
    (0): Linear(in_features=784, out_features=500, bias=True)
    (1): ReLU()
    (2): Linear(in_features=500, out_features=500, bias=True)
    (3): ReLU()
  )
  (output_layer): Sequential(
    (0): Linear(in_features=500, out_features=10, bias=True)
  )
)

 Number of parameters in the model (weights and biases):  648010
```


Mila

Tutorial Questions



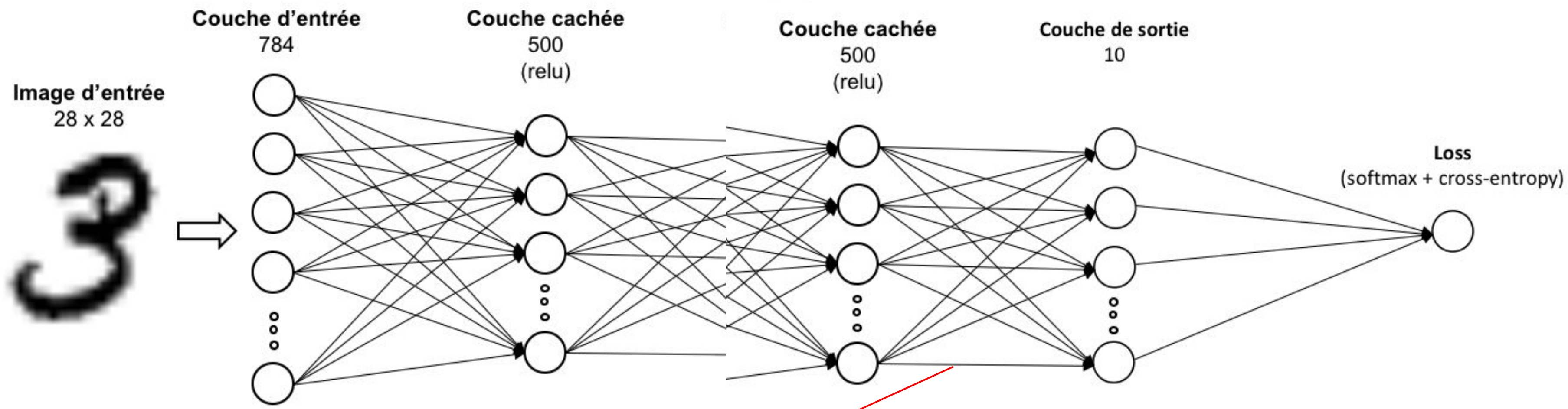

```
MLP(
  (hidden_layer): Sequential(
    (0): Linear(in_features=784, out_features=500, bias=True)
    (1): ReLU()
    (2): Linear(in_features=500, out_features=500, bias=True)
    (3): ReLU()
  )
  (output_layer): Sequential(
    (0): Linear(in_features=500, out_features=10, bias=True)
  )
)

Number of parameters in the model (weights and biases):  648010
```

2nd hidden layer:

500 vectors of 1x500 + 500 biases = 500*501 = 250 500

500 to 500 - 2nd layer

Tutorial Questions



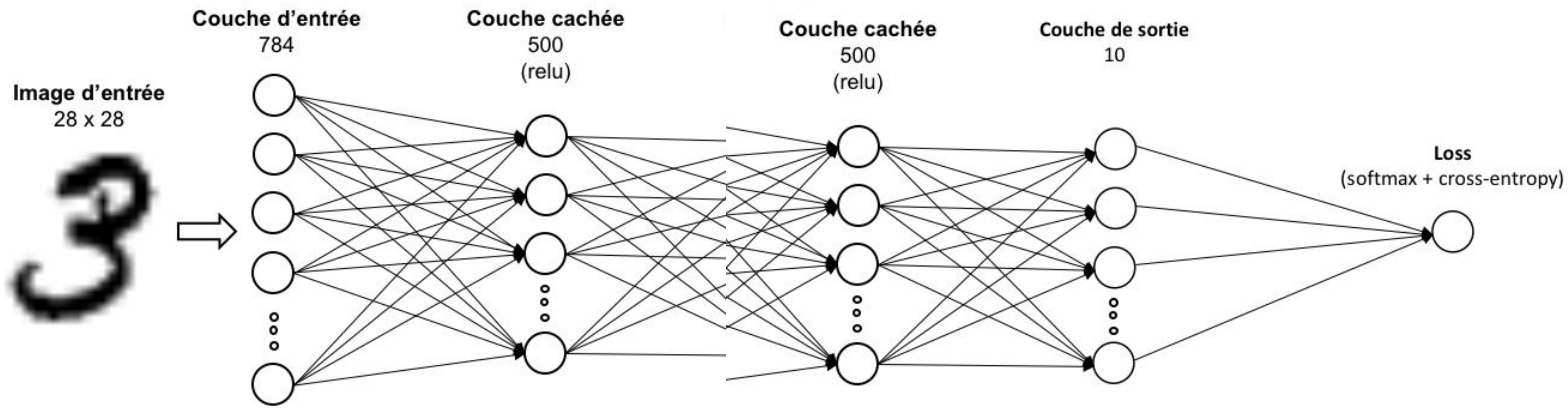

```
MLP(
  (hidden_layer): Sequential(
    (0): Linear(in_features=784, out_features=500, bias=True)
    (1): ReLU()
    (2): Linear(in_features=500, out_features=500, bias=True)
    (3): ReLU()
  )
  (output_layer): Sequential(
    (0): Linear(in_features=500, out_features=10, bias=True)
  )
)

Number of parameters in the model (weights and biases):  648010
```

500 to 10 - output layer

Output layer:

10 vectors of 1x500 + 10 biases = 501*10 = 5010

Tutorial Questions



1st layer + 2nd layer + 3rd layer =
392 500 + 250 500 + 5010 = 648 010

```
MLP(
  (hidden_layer): Sequential(
    (0): Linear(in_features=784, out_features=500, bias=True)
    (1): ReLU()
    (2): Linear(in_features=500, out_features=500, bias=True)
    (3): ReLU()
  )
  (output_layer): Sequential(
    (0): Linear(in_features=500, out_features=10, bias=True)
  )
)
```

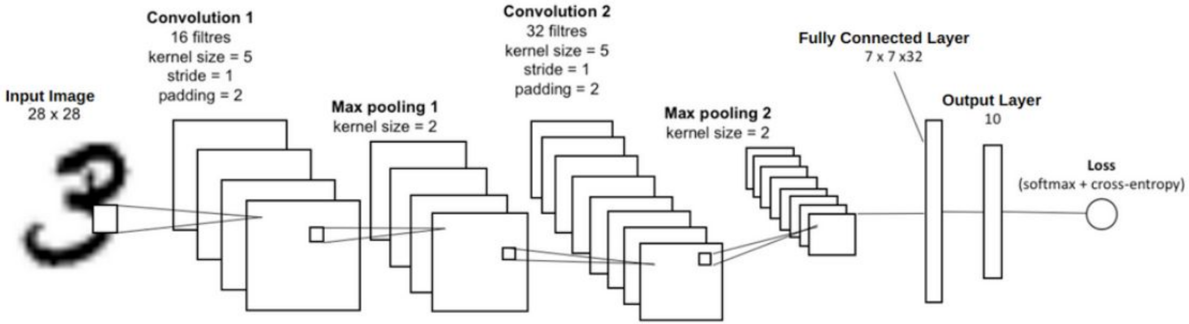
```
Number of parameters in the model (weights and biases):  648010
```

Tutorial Questions

Question 6

1 point possible (graded)

Consider the following architecture used on MNIST. We have 10 classes and input images of size 28x28.



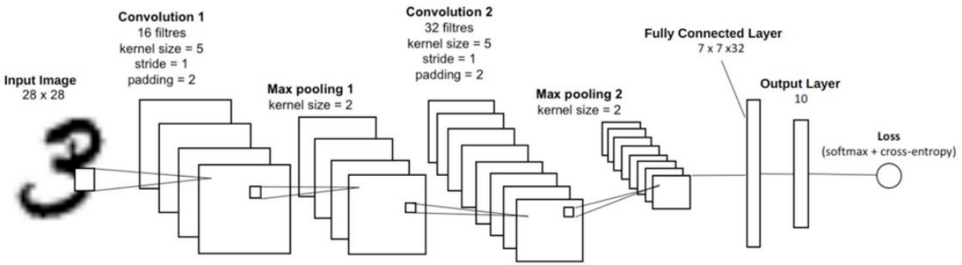
How many trainable parameters do we have in total?

Tutorial Questions

Question 6

1 point possible (graded)

Consider the following architecture used on MNIST. We have 10 classes and input images of size 28x28.



How many trainable parameters do we have in total?

1st layer: 1 input channel with 16 kernels (5x5):

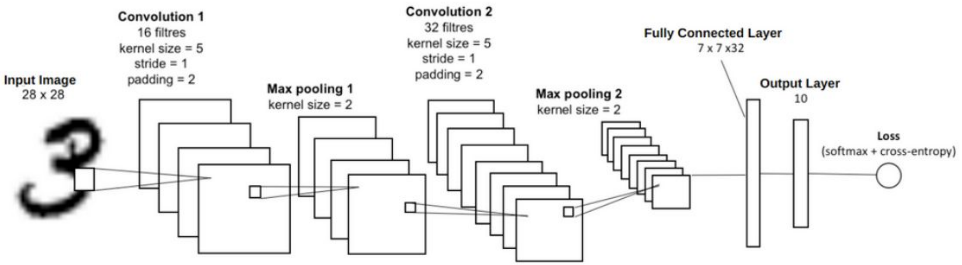
16 x (5x5) + 16 biases = 416

Tutorial Questions

Question 6

1 point possible (graded)

Consider the following architecture used on MNIST. We have 10 classes and input images of size 28x28.



How many trainable parameters do we have in total?

2nd layer: 16 input channels with 32 output channels:

[16 x (5x5)] x 32 + 32 biases = 12 832

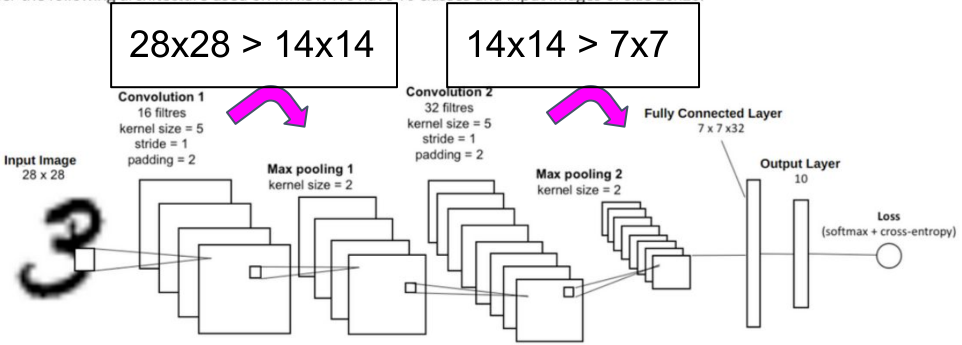
Mila

Tutorial Questions

Question 6

1 point possible (graded)

Consider the following architecture used on MNIST. We have 10 classes and input images of size 28x28.



How many trainable parameters do we have in total?

Fully connected layer - why 7x7x32? Because 28 / 2 / 2 = 7 (max pooling reduces our input, padding makes this a “same” convolution).

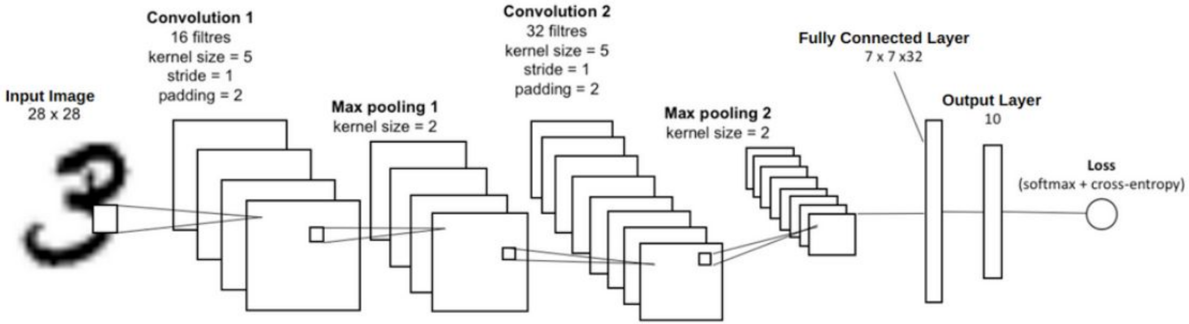
So our feature maps are **HxW = 7x7, and depth = 32**. Flattened, that is equivalent to 7x7x32 inputs. We want 10 outputs: (7x7x32) x 10 + 10 biases = 15 690

Tutorial Questions

Question 6

1 point possible (graded)

Consider the following architecture used on MNIST. We have 10 classes and input images of size 28x28.



How many trainable parameters do we have in total?

416 + 12 832 + 15 690 = 28 938