*Pole Universitaire Léonard De Vinci*
*ESILV, 2023/2024*

**Deep Learning**

**Duration** :1 h 30 - 22/02/2023
**The use of a calculator is recommended**

## Before starting

Please notice that you can add your answers on the exam statement sheet without forgetting to return it after adding your first and last names.

## Exercice 1 (7.5 points)

We trained using the Keras API, the following model on the CIFAR 10 data :

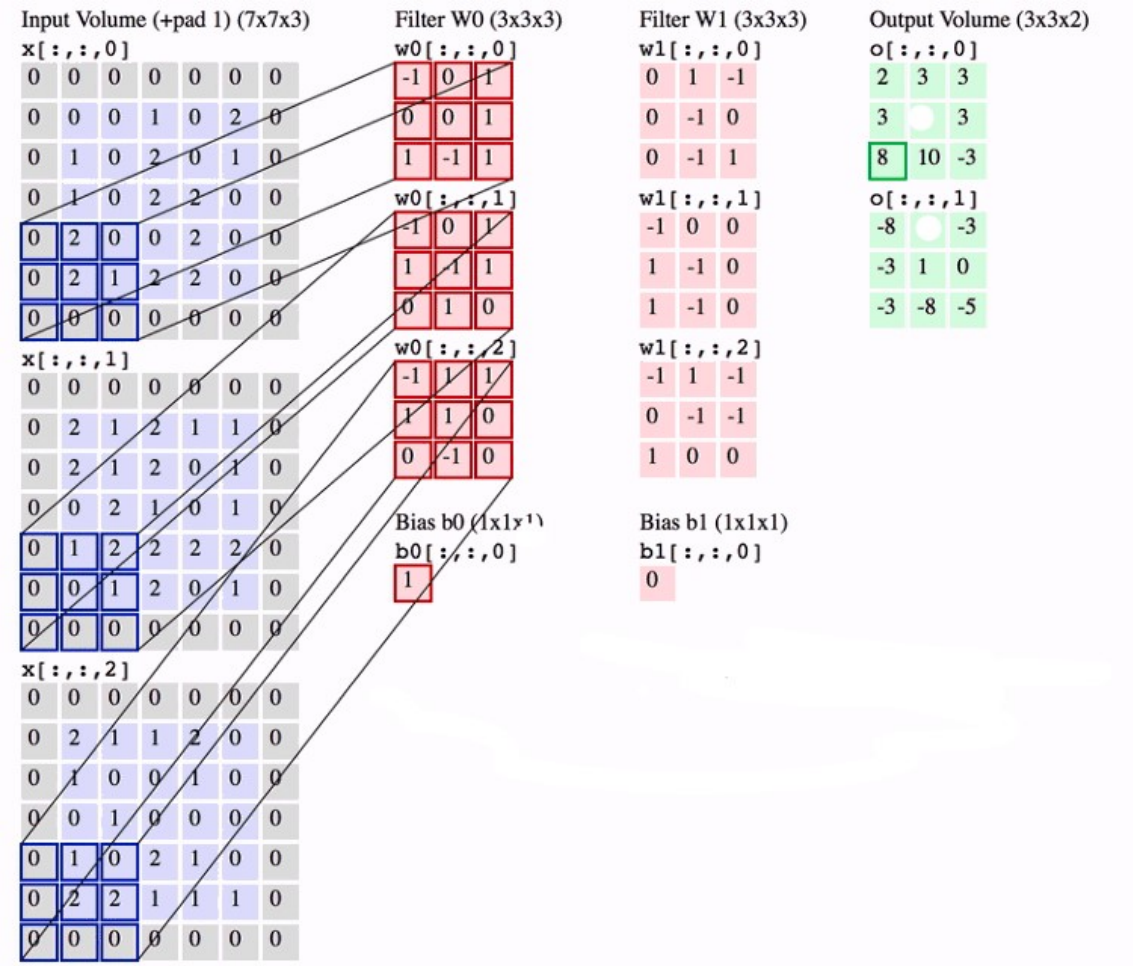
```
model = keras.Sequential([
    keras.layers.Conv2D(4, kernel_size=3, activation='relu', padding = 'same', input_shape = (32, 32, 3)),
    keras.layers.MaxPooling2D(pool_size= (2,2), strides=2),
    keras.layers.Conv2D(8, kernel_size=3, padding = 'same', activation='relu'),
    keras.layers.MaxPooling2D(pool_size = (2,2), strides= 2),
    keras.layers.Flatten(),
    keras.layers.Dense(28, activation = 'relu'),
    keras.layers.Dense(10, activation = 'softmax')
])
```

Supposing that all neurons have an intercept( a constant $w_0$ weight), complete the following table by adding the missed output shape and the number of parameters for the used layers.In total, you have to add 15 integers values. Note that the given total number of parameters may help you.

```
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 conv2d_2 (Conv2D)            (      32, 32, 4)         112

 max_pooling2d_2 (MaxPooling  (     ,  ,  )            0
 2D)

 conv2d_3 (Conv2D)            (     ,  ,  )

 max_pooling2d_3 (MaxPooling  (     ,  ,  )            0
 2D)

 flatten_1 (Flatten)          (     )                  0

 dense_2 (Dense)              (     )

 dense_3 (Dense)              (     )

=================================================================
Total params: 15,062
Trainable params: 15,062
Non-trainable params: 0
```

## Exercice 2 (2 points)

Complete the following convolutionnal operation by computing the two missed values in the output tensor.




## Exercice 3 (4.5 points)

1. We show here a code of an LSTM applied on yahoo database. Complete the missed two *return_sequences* values that makes the network working correctly.

```
from tensorflow.keras.layers import Dense, Activation, Dropout
from tensorflow.keras.layers import LSTM
from tensorflow.keras.models import Sequential
import time

model = Sequential()

model.add(LSTM(input_shape=(look_back,1), units=100, return_sequences= · ))
model.add(Dropout(0.2))

model.add(LSTM(units=50, return_sequences= · ))
model.add(Dropout(0.2))

model.add(Dense(units=1, activation ='linear'))

model.summary()
```

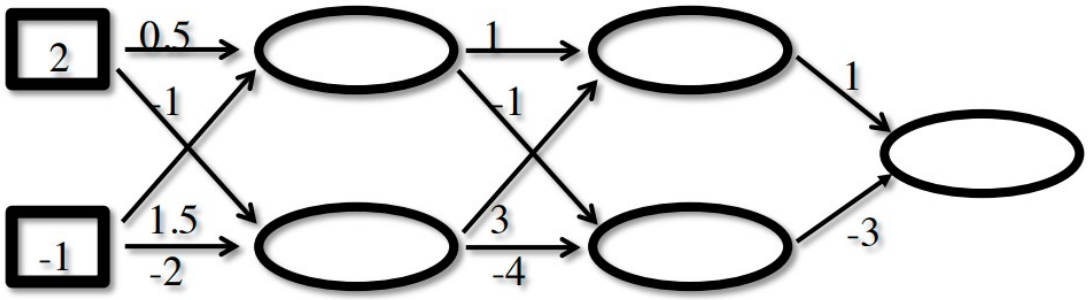
2. We show now a code of an autoencoder on the MNIST handwritten digit database. Complete the missed three filter numbers at the definition of the con2D layers in order to recover the correct dimension.

```
x = Conv2D(16, (3, 3), activation='relu', padding='same')(input_img)
x = MaxPooling2D((2, 2), padding='same')(x)
x = Conv2D(8, (3, 3), activation='relu', padding='same')(x)
x = MaxPooling2D((2, 2), padding='same')(x)
x = Conv2D(8, (3, 3), activation='relu', padding='same')(x)
encoded = MaxPooling2D((2, 2), padding='same')(x)

# at this point the representation is (4, 4, 8) i.e. 128-dimensional

x = Conv2D( , (3, 3), activation='relu', padding='same')(encoded)
x = UpSampling2D((2, 2))(x)
x = Conv2D( , (3, 3), activation='relu', padding='same')(x)
x = UpSampling2D((2, 2))(x)
x = Conv2D( , (3, 3), activation='relu')(x)
x = UpSampling2D((2, 2))(x)
decoded = Conv2D(1, (3, 3), activation='sigmoid', padding='same')(x)

autoencoder = Model(input_img, decoded)
```

Note that model summary given here may help you :

| Layer (type) | Output Shape | Param # |
|---|---|---|
| input_1 (InputLayer) | [(None, 28, 28, 1)] | 0 |
| conv2d (Conv2D) | (None, 28, 28, 16) | 160 |
| max_pooling2d (MaxPooling2D) | (None, 14, 14, 16) | 0 |
| conv2d_1 (Conv2D) | (None, 14, 14, 8) | 1160 |
| max_pooling2d_1 (MaxPooling2D) | (None, 7, 7, 8) | 0 |
| conv2d_2 (Conv2D) | (None, 7, 7, 8) | 584 |
| max_pooling2d_2 (MaxPooling2D) | (None, 4, 4, 8) | 0 |
| conv2d_3 (Conv2D) | (None, 4, 4, 8) | 584 |
| up_sampling2d (UpSampling2D) | (None, 8, 8, 8) | 0 |
| conv2d_4 (Conv2D) | (None, 8, 8, 8) | 584 |
| up_sampling2d_1 (UpSampling2D) | (None, 16, 16, 8) | 0 |
| conv2d_5 (Conv2D) | (None, 14, 14, 16) | 1168 |
| up_sampling2d_2 (UpSampling2D) | (None, 28, 28, 16) | 0 |
| conv2d_6 (Conv2D) | (None, 28, 28, 1) | 145 |

## Exercice 4 (6 points)




We consider the following neural network and want to show all steps of the back-propagation algorithm with the imput $x := [2, -1]$ and output $y = 0$. We suppose that there is no intercept in this network and that the sigmoid is the only used activaion function :

$$\sigma(x) := \frac{1}{1 + \exp(-x)}$$

1. Propagate the imput foword and fill the cells with the activation value of the neuron (recall the used expression).

2. Compute the delta at the last neuron and propagate it backward to all neurons by writing above the cells the obtained $\delta$ values.
3. Now we suppose that we use the Relu activation function :

$$Relu(x) := \max(x, 0)$$

Propagate again the imput foword and fill the cells with the activation value of the neuron (use the second plot of the network). Compare the error with respect to the first case.