An Introduction to Deep Learning With Python

[2.2] Data representations for Neural Networks

Prof. Yuzo Iano

pgs: 31 - 34

Scalars (0D Tensors)

```
In [1]:  import numpy as np
         x = np.array(12)
         x
```

```
Out[1]:  array(12)
```

```
In [2]:  x.ndim
```

```
Out[2]:  0
```

Vectors (1D Tensors)

```
In [3]:  x = np.array([12, 3, 6, 14, 7])
         x
```

```
Out[3]:  array([12,  3,  6, 14,  7])
```

```
In [4]:  x.ndim
```

```
Out[4]:  1
```

Matrices (2D Tensors)

```
In [5]:  x = np.array([[5, 78, 2, 34, 0],
                       [6, 79, 3, 35, 1],
                       [7, 80, 4, 36, 2]])
         x.ndim
```

```
Out[5]:  2
```

3D tensors and higher-dimensional tensors

In [6]:

```
x = np.array([[[5, 78, 2, 34, 0],
               [6, 79, 3, 35, 1],
               [7, 80, 4, 36, 2]],
               [[5, 78, 2, 34, 0],
               [6, 79, 3, 35, 1],
               [7, 80, 4, 36, 2]],
               [[5, 78, 2, 34, 0],
               [6, 79, 3, 35, 1],
               [7, 80, 4, 36, 2]]])
x.ndim
```

Out[6]:

```
3
```

Key attributes

In [7]:

```
from keras.datasets import mnist

(train_images, train_labels),(test_images, test_labels) = mnist.load_data()
print('ndim = ', train_images.ndim)
print('Train Shape = ', train_images.shape)
print('dtype = ', train_images.dtype)
```

```
Using TensorFlow backend.

ndim =  3
Train Shape =  (60000, 28, 28)
dtype =  uint8
```

Displaying the fourth digit

In [8]:

```
digit = train_images[4]

import matplotlib.pyplot as plt
plt.imshow(digit, cmap = plt.cm.binary)
plt.show()
```

```
<Figure size 640x480 with 1 Axes>
```

Manipulating tensors in Numpy

In [9]:

```
my_slice = train_images[10: 100]
print('my_slice shape = ', my_slice.shape)
```

```
my_slice shape =  (90, 28, 28)
```

In [10]:

```
my_slice = train_images[10: 100, 0:28, 0:28]
print('my_slice shape = ', my_slice.shape)
```

```
my_slice shape =  (90, 28, 28)
```

In [11]:

```
my_slice = train_images[:, 14:, 14:]
```

In [12]:

```
my_slice = train_images[:, 7:-7, 7:-7]
```

The notion of data batches

```
In [13]: batch = train_images[:128]
         batch.shape

Out[13]: (128, 28, 28)

In [14]: batch = train_images[128:256]
         batch.shape

Out[14]: (128, 28, 28)

In [15]: n = 14
         batch = train_images[128 * n:128 * (n + 1)]
         batch.shape

Out[15]: (128, 28, 28)
```

Pablo Minango

- pablodavid218@gmail.com