**CSC-40070 Applications of AI, Machine Learning and Data Science**
**Lab 11: Image recognition using deep learning networks**

## Task 1: Build and train a Deep Convolutional Neural Network inspired by LeNet-5 to classify images from the MNIST database of handwritten digits.

As in the previous labs, use TensorFlow and Keras. Import the following dependencies:

```
import tensorflow
from tensorflow.keras.datasets import mnist
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout # Dropout is optional
from tensorflow.keras.layers import Flatten, Conv2D, MaxPooling2D
```

The MNIST database consists of 60,000 training pairs and 10,000 validation pairs, with each pair being a 28 pixel * 28 pixel 8-bit image of a handwritten digit and its classification (0 to 9). You can load it like this:

```
(X_train, y_train), (X_valid, y_valid) = mnist.load_data()
```

The next step is to pre-process the data:

- Reshape X_train to (60000, 28, 28, 1) and X_valid to (10000, 28, 28, 1), as Conv2D requires the extra dimension to allow for multi-channel input (e.g. 3 for RGB), even if you only have one.
- Also convert each pixel value from an integer (between 0 and 255, as 8-bit) to be a floating-point value ('float32') between 0 and 1.
- Use the “to_categorical” method to reshape y_train to (60000, 10) and y_valid to (10000, 10), containing one 1 per row, in the corresponding column, and 0s in all other entries.

Create a deep convolutional neural network with the following parameters:

- hint: start with `model = Sequential()` and then use a sequence of `model.add()` calls
- Layer 1: 2D convolution layer with 32 3*3 filters, input_shape=(28, 28, 1), ReLU
- Layer 2: 2D convolution layer with 64 3*3 filters, ReLU
- Layer 3: 2D max-pooling layer with 2*2 pool size
- Use `model.add(Flatten())` ahead of the following Dense layer
- Layer 4: Regular fully-connected (aka densely-connected) NN layer with 128 neurons
- Layer 5: Classification layer: model.add(Dense(10, activation='softmax'))

Finally configure and train the model, e.g.:

```
model.compile(loss='categorical_crossentropy', optimizer='nadam', \
              metrics=['accuracy'])
model.fit(X_train, y_train, batch_size=128, epochs=10, verbose=1, \
          validation_data=(X_valid, y_valid))
```

When you run your model, you should see validation accuracy increase from approx. 10% (1 in 10) to approx. 99%. (Optionally, you can try using “Dropout” in your code to improve accuracy slightly.)

**Solution**: Jupyter Notebook (view in nbviewer) **run in Google Colab**
In Google Colab, start by changing Runtime type to GPU to speed up training (revert to None/CPU if no GPU available); click in the first section and repeatedly press Shift+Return to run each section in sequence.

Credit: https://github.com/jonkrohn/DLTFpT/blob/master/notebooks/lenet_in_tensorflow.ipynb

Jon Krohn's textbook: Deep Learning Illustrated (follow that link to order as a PDF via InformIT)

P.T.O.

**Task 2: Modify your Deep Convolutional Neural Network (or the example solution) to classify images from the Fashion-MNIST database of clothing articles; explore the data, train the network and perform inference.**

Change `import mnist` to `import fashion_mnist` and `mnist.load_data()` to `fashion_mnist.load_data()`.

You can display the first images in the training set as follows:

```
from matplotlib import pyplot as plt

plt.figure(figsize=(5,5))
for k in range(12):
    plt.subplot(3, 4, k+1)
    plt.imshow(X_train[k], cmap='Greys')
    plt.axis('off')
plt.tight_layout()
plt.show()
```

The first image in the validation is an "ankle boot", category 9:

```
plt.imshow(X_valid[0], cmap='Greys')
y_valid[0]
9 # output
```

Once you have trained your neural network, use it to perform inference. For example, using the first image in the validation set:

```
valid_0 = X_valid[0].reshape(1, 28, 28, 1)
model.predict(valid_0) # shows output per category
model.predict_classes(valid_0)
array([9]) # correct output
```

**Solution**: Jupyter Notebook (view in nbviewer) **run in Google Colab**
In Google Colab, change the number of epochs from 1 to e.g. 10, before you run the code.

Credit: https://github.com/jonkrohn/DLTFpT/blob/master/notebooks/lenet_in_tensorflow_for_fashion_MNIST.ipynb