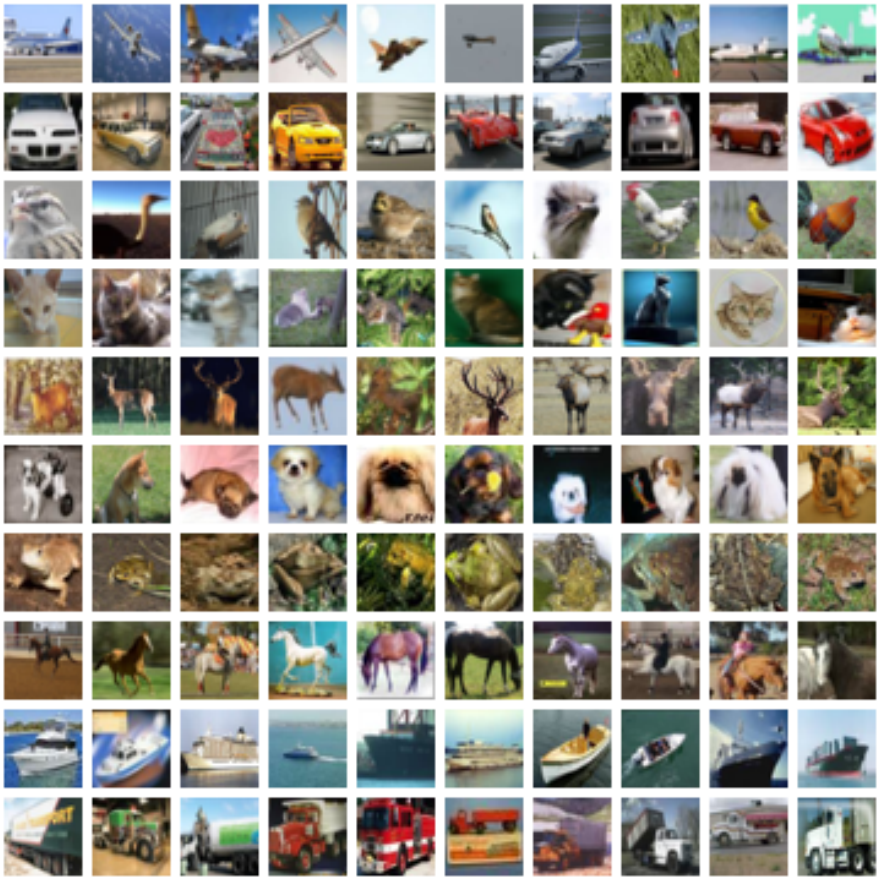


COMP47590

Advanced Machine Learning

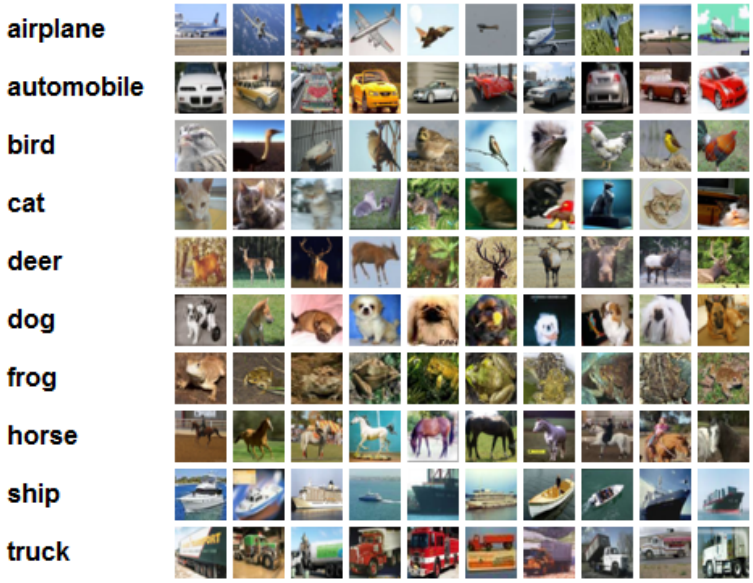
Lab Task 3: Deep Learning With Keras

Introduction

In this workshop we will experiment with deep neural network models in Python using the keras library.

Tasks

1. The cifar10 dataset contains 60000 32x32 colour images of ten different types of objects - 6000 images per object. This is used as a benchmark for classifying the contents of an image.



Keras ships with the cifar10 dataset included and it can be loaded into a training and test set as follows:

```
keras.datasets import cifar10
(x_train, y_train), (x_test, y_test) =
cifar10.load_data()
```

The following code block draws 10 random images from cifar10 and displays their classes

```
classes = {0:"airplane", 1:"automobile",
2:"bird",3:"cat", 4:"deer",
5:"dog", 6:"frog", 7:"horse", 8:"ship", 9:"truck"}
for i in range(0, 9):
    i_rand = randint(0, x_train.shape[0])
    print(classes[y_train[i_rand][0]])
    pyplot.imshow(toimage(x_train[i_rand]))
    pyplot.show()
```

Use this code block to examine some examples of cifar10 images.

2. Build a simple feedforward neural network to classify images into the 10 classes. To do this flatten the input images into a 3072 input feature vector using the following code:

```
x_train_flat = x_train.reshape(x_train.shape[0], 3072)
x_test_flat = x_test.reshape(x_test.shape[0], 3072)
```

```
x_train_flat = x_train_flat.astype('float32')
x_test_flat = x_test_flat.astype('float32')
x_train_flat /= 255
x_test_flat /= 255
```

3. Evaluate the performance of the feed-forward network on training and test sets.

4. Experiment with different feed-forward model sizes and different hyper parameters.

5. Build a convolutional network to do the classification job. To handle the 32 x 32 colour images the first layer in the convolutional network should look like this

```
model = Sequential()
model.add(Conv2D(32, (3, 3), padding='same',
                 input_shape=x_train.shape[1:]))
```

   and the last layer should look like this

```
model.add(Dense(num_classes))
model.add(Activation('softmax'))
```

   Experiment with combinations of 2D convolutions, max-pooling and dropout.

   A flatten layer will be required to transition from convolutional and max pooling layers to dense layers.

6. Evaluate the performance of the convolutional network on training and test sets.

7. Experiment with different convolutional model sizes and different hyper parameters.