```
import time
import matplotlib.pyplot as plt
import numpy as np
from keras.models import Sequential
from keras.layers.convolutional import Convolution2D, MaxPooling2D
from keras.layers import Activation, Flatten, Dense, Dropout
from keras.layers.normalization import BatchNormalization
from keras.utils import np_utils
```

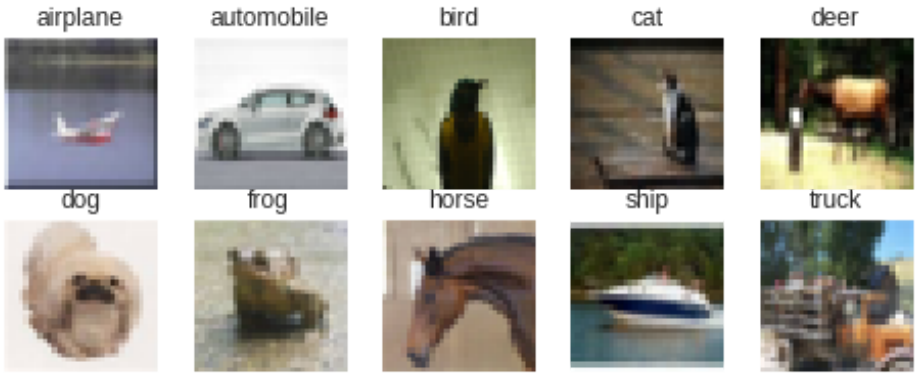
```
Using TensorFlow backend.
```

```
from keras import backend as K
if K.backend()=='tensorflow':
  K.set_image_dim_ordering("th")
```

```
% matplotlib inline
np.random.seed(42)
```

```
from keras.datasets import cifar10
(x_train, x_labels), (y_test, y_labels) = cifar10.load_data()
num_train, img_channels, img_rows, img_cols = x_train.shape
num_test, _, _, _ = y_test.shape
num_classes = len(np.unique(x_labels))
```

```
class_names = ['airplane','automobile','bird','cat','deer','dog','frog','horse','ship','tru
fig = plt.figure(figsize=(8,3))
for i in range(num_classes):
  ax = fig.add_subplot(2, 5, 1 + i, xticks=[], yticks=[])
  idx = np.where(x_labels[:]==i)[0]
  features_idx = x_train[idx,::]
  img_num = np.random.randint(features_idx.shape[0])
  im = np.transpose(features_idx[img_num,::], (1, 2, 0))
  ax.set_title(class_names[i])
  plt.imshow(im)
plt.show()
```




```
train_features = x_train.astype('float32')/255
test_features = y_test.astype('float32')/255
# convert class labels to binary class labels
train_labels = np_utils.to_categorical(x_labels, num_classes)
test_labels = np_utils.to_categorical(y_labels, num_classes)
```

```
def accuracy(test_x, test_y, model):
  result = model.predict(test_x)
  predicted_class = np.argmax(result, axis=1)
  true_class = np.argmax(test_y, axis=1)
  num_correct = np.sum(predicted_class == true_class)
  accuracy = float(num_correct)/result.shape[0]
  return (accuracy * 100)


model = Sequential()
model.add(Convolution2D(48, 3, 3, border_mode='same', input_shape=(3, 32, 32)))
model.add(Activation('relu'))
model.add(Convolution2D(48, 3, 3))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.5))
model.add(Convolution2D(96, 3, 3, border_mode='same'))
model.add(Activation('relu'))
model.add(Convolution2D(96, 3, 3))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.5))
model.add(Convolution2D(192, 3, 3, border_mode='same'))
model.add(Activation('relu'))
model.add(Convolution2D(192, 3, 3))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.5))
model.add(Flatten())
model.add(Dense(512))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(256))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax'))
# Compile the model
model.compile(optimizer='adam', loss='poisson', metrics=['accuracy'])
# Train the model
start = time.time()
model_info = model.fit(train_features, train_labels,
batch_size=50, nb_epoch=100,
validation_data = (test_features, test_labels),
verbose=0)
end = time.time()
print ("Accuracy on test data is: %0.2f"%accuracy(test_features, test_labels, model))
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2: UserWarning: Update y

/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:4: UserWarning: Update y
  after removing the cwd from sys.path.
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:8: UserWarning: Update y

/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:10: UserWarning: Update
  # Remove the CWD from sys.path while we load stuff.
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:14: UserWarning: Update

/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:16: UserWarning: Update
  app.launch_new_instance()
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:35: UserWarning: The `n
Accuracy on test data is: 74.86
```