In [1]:

```
from keras.datasets import cifar10
from keras.utils import np_utils
from keras.models import Sequential
from keras.layers.core import Dense, Dropout, Activation, Flatten
from keras.layers.convolutional import Conv2D, MaxPooling2D
from keras.optimizers import SGD, Adam, RMSprop
import matplotlib.pyplot as plt
```

```
Using TensorFlow backend.
```

In [2]:

```
# CIFAR-10 содержит 60K изображений 32*32*3 канала цвета
IMG_CHANNELS = 3
IMG_ROWS = 32
IMG_COLS = 32

BATCH_SIZE = 128
NB_EPOCH = 25
NB_CLASSES = 10
VERBOSE = 1
VALIDATION_SPLIT = 0.2
OPTIM = RMSprop()
```

In [3]:

```
(X_train, y_train), (X_test, y_test) = cifar10.load_data()
```

```
Downloading data from https://www.cs.toronto.edu/~kriz/cifar-10-pyth
on.tar.gz
170500096/170498071 [==============================] - 18s 0us/step
```

In [4]:

```
# Пребобразуем к категориальному виду
Y_train = np_utils.to_categorical(y_train, NB_CLASSES)
Y_test = np_utils.to_categorical(y_test, NB_CLASSES)
```

In [5]:

```
# Преобразуем к формату с плавующей точкой и нормируем к диапазоу (0,1)
X_train = X_train.astype('float32')
X_test = X_test.astype('float32')
X_train /= 255
X_test /= 255
```

In [6]:

```
# Описываем нейросеть
model = Sequential()
model.add(Conv2D(32, (3, 3),padding='same',
    input_shape=(IMG_ROWS,IMG_COLS, IMG_CHANNELS)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(512))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(NB_CLASSES))
model.add(Activation('softmax'))
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv2d_1 (Conv2D)            (None, 32, 32, 32)        896       
_________________________________________________________________
activation_1 (Activation)    (None, 32, 32, 32)        0         
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 16, 16, 32)        0         
_________________________________________________________________
dropout_1 (Dropout)          (None, 16, 16, 32)        0         
_________________________________________________________________
flatten_1 (Flatten)          (None, 8192)              0         
_________________________________________________________________
dense_1 (Dense)              (None, 512)               4194816   
_________________________________________________________________
activation_2 (Activation)    (None, 512)               0         
_________________________________________________________________
dropout_2 (Dropout)          (None, 512)               0         
_________________________________________________________________
dense_2 (Dense)              (None, 10)                5130      
_________________________________________________________________
activation_3 (Activation)    (None, 10)                0         
=================================================================
Total params: 4,200,842
Trainable params: 4,200,842
Non-trainable params: 0
_________________________________________________________________
```

In [7]:

```
# Обучение модели
model.compile(loss='categorical_crossentropy', optimizer=OPTIM,
    metrics=['accuracy'])
model.fit(X_train, Y_train, batch_size=BATCH_SIZE,
    epochs=NB_EPOCH, validation_split=VALIDATION_SPLIT,
    verbose=VERBOSE)

score = model.evalaute(X_test, Y_test,
    batch_size=BATCH_SIZE, verbose=VERBOSE)
print("Test score:", score[0])
print("Test accuracy", score[1])
```

```
Train on 40000 samples, validate on 10000 samples
Epoch 1/25
40000/40000 [==============================] - 65s 2ms/step - loss:
1.7699 - accuracy: 0.3795 - val_loss: 1.4241 - val_accuracy: 0.4981
Epoch 2/25
40000/40000 [==============================] - 67s 2ms/step - loss:
1.3928 - accuracy: 0.5049 - val_loss: 1.2984 - val_accuracy: 0.5457
Epoch 3/25
40000/40000 [==============================] - 58s 1ms/step - loss:
1.2580 - accuracy: 0.5574 - val_loss: 1.3014 - val_accuracy: 0.5435
Epoch 4/25
40000/40000 [==============================] - 59s 1ms/step - loss:
1.1651 - accuracy: 0.5885 - val_loss: 1.1644 - val_accuracy: 0.5897
Epoch 5/25
40000/40000 [==============================] - 64s 2ms/step - loss:
1.0920 - accuracy: 0.6165 - val_loss: 1.0723 - val_accuracy: 0.6293
Epoch 6/25
40000/40000 [==============================] - 75s 2ms/step - loss:
1.0340 - accuracy: 0.6382 - val_loss: 1.0436 - val_accuracy: 0.6409
Epoch 7/25
40000/40000 [==============================] - 65s 2ms/step - loss:
0.9777 - accuracy: 0.6564 - val_loss: 1.0599 - val_accuracy: 0.6352
Epoch 8/25
40000/40000 [==============================] - 76s 2ms/step - loss:
0.9364 - accuracy: 0.6765 - val_loss: 1.0585 - val_accuracy: 0.6378
Epoch 9/25
40000/40000 [==============================] - 70s 2ms/step - loss:
0.8890 - accuracy: 0.6901 - val_loss: 1.0389 - val_accuracy: 0.6496
Epoch 10/25
40000/40000 [==============================] - 82s 2ms/step - loss:
0.8493 - accuracy: 0.7051 - val_loss: 1.0453 - val_accuracy: 0.6549
Epoch 11/25
40000/40000 [==============================] - 69s 2ms/step - loss:
0.8166 - accuracy: 0.7176 - val_loss: 1.0171 - val_accuracy: 0.6647
Epoch 12/25
40000/40000 [==============================] - 65s 2ms/step - loss:
0.7880 - accuracy: 0.7279 - val_loss: 0.9896 - val_accuracy: 0.6724
Epoch 13/25
40000/40000 [==============================] - 65s 2ms/step - loss:
0.7516 - accuracy: 0.7411 - val_loss: 0.9797 - val_accuracy: 0.6774
Epoch 14/25
40000/40000 [==============================] - 65s 2ms/step - loss:
0.7297 - accuracy: 0.7485 - val_loss: 1.0707 - val_accuracy: 0.6528
Epoch 15/25
40000/40000 [==============================] - 65s 2ms/step - loss:
0.7020 - accuracy: 0.7566 - val_loss: 1.0691 - val_accuracy: 0.6576
Epoch 16/25
40000/40000 [==============================] - 65s 2ms/step - loss:
0.6790 - accuracy: 0.7636 - val_loss: 1.0238 - val_accuracy: 0.6731
Epoch 17/25
40000/40000 [==============================] - 65s 2ms/step - loss:
0.6649 - accuracy: 0.7706 - val_loss: 1.0035 - val_accuracy: 0.6784
Epoch 18/25
40000/40000 [==============================] - 65s 2ms/step - loss:
0.6387 - accuracy: 0.7798 - val_loss: 1.0745 - val_accuracy: 0.6749
Epoch 19/25
40000/40000 [==============================] - 62s 2ms/step - loss:
0.6175 - accuracy: 0.7865 - val_loss: 1.0340 - val_accuracy: 0.6711
Epoch 20/25
40000/40000 [==============================] - 63s 2ms/step - loss:
0.5965 - accuracy: 0.7962 - val_loss: 0.9977 - val_accuracy: 0.6826
```

```
Epoch 21/25
40000/40000 [==============================] - 63s 2ms/step - loss:
0.5835 - accuracy: 0.7996 - val_loss: 1.0935 - val_accuracy: 0.6769
Epoch 22/25
40000/40000 [==============================] - 63s 2ms/step - loss:
0.5634 - accuracy: 0.8073 - val_loss: 1.0394 - val_accuracy: 0.6700
Epoch 23/25
40000/40000 [==============================] - 64s 2ms/step - loss:
0.5508 - accuracy: 0.8134 - val_loss: 1.1696 - val_accuracy: 0.6801
Epoch 24/25
40000/40000 [==============================] - 63s 2ms/step - loss:
0.5410 - accuracy: 0.8146 - val_loss: 1.1272 - val_accuracy: 0.6880
Epoch 25/25
40000/40000 [==============================] - 64s 2ms/step - loss:
0.5254 - accuracy: 0.8209 - val_loss: 1.1052 - val_accuracy: 0.6722


---------------------------------------------------------------------
-------
AttributeError                            Traceback (most recent cal
l last)
<ipython-input-7-217e1318fc24> in <module>
      6     verbose=VERBOSE)
      7
----> 8 score = model.evalute(X_test, Y_test,
      9     batch_size=BATCH_SIZE, verbose=VERBOSE)
     10 print("Test score:", score[0])

AttributeError: 'Sequential' object has no attribute 'evalute'
```

In [ ]:

```
model_json = model.to_json()
open('cifar10_arch.json','w').write(model_json)
model,save_weights('cifar10_weights.h5', overwrite = True)
```

In [ ]:

```

```

In [ ]:

```

```