# **8.2. Использование предварительно обученных нейронных сетей (сверточная сеть)**

# 

( <https://www.asozykin.ru/courses/nnpython> )

Keras позволяет не только обучать глубокие нейронные сети, но и содержит несколько готовых предварительно обученных сетей. Эти сети можно использовать, не тратя время и вычислительные ресурсы на обучение.

В Keras предварительно обученные сети находятся в [модуле applications](https://keras.io/applications/). Большая часть сетей предназначена для распознавания объектов на изображениях из набора данных [ImageNet](http://image-net.org/), но есть и сеть для обработки музыки. Модуль содержит следующие сети:

1. VGG16 - сеть [Visual Geometry Group из университета Оксфорда для распознавания объектов на изображениях](http://www.robots.ox.ac.uk/~vgg/research/very_deep/), состоит из 16 слоев.
2. VGG19 - еще одна сеть Visual Geometry Group для распознавания объектов, но содержит 19 слоев.
3. Inception v3 - [нейронная сеть компании Google](http://arxiv.org/abs/1512.00567) для распознавания объектов на изображениях.
4. ResNet50 - [нейронная сеть компании Microsoft](https://arxiv.org/abs/1512.03385), использующая остаточное обучение (residual learning). Применяется для распознавании объектов на изображениях.
5. Xception - [модификация сети Inception](https://arxiv.org/abs/1610.02357) от создателя Keras François Chollet.
6. CRNN for music tagging - сверточная рекуррентная [нейронная сеть для классификации музыки](https://arxiv.org/abs/1609.04243).

Как можно видеть, сети очень глубокие, содержат десятки слоев. Для их обучения требуются огромные вычислительные мощности, которые есть только у таких крупных компаний, как Google и Microsoft. Но благодаря тому, что компании выкладывают обученные сети в открытый доступ, вы можете использовать их в своих программах на Keras. Давайте посмотрим, как это сделать для распознавания объектов на изображениях.

Набор данных ImageNet

Начнем с набора данных [ImageNet](http://image-net.org/), который используется для обучения большинства нейронных сетей в Keras. Это открытый набор данных из 14 миллионов изображений. Для каждого изображения указано, какой объект на нем. Таким образом, ImageNet можно использовать для обучения с учителем.

ImageNet содержит двадцать одну тысячу классов объектов, однако нейронные сети обучают распознавать только 1000 из них (что очень много по сравнению с [CIFAR-10](https://www.asozykin.ru/courses/nnpython-lab2) и другими аналогичными наборами изображений). В ImageNet классы образуют иерархию. Например, указывается, что на изображении не просто кот, а конкретный вид кота (сиамский, египетский и т.п.).

Архитектура сети VGG16

VGG16 - это сверточная нейронная сеть, которая содержит 16 слоев. Сеть состоит из двух частей:

1. Первая часть сети выделяет **характерные признаки** в изображении. Состоит из чередующихся каскадов свертки и подвыборки. Сначала идут два каскада свертка-свертка-подвыборка, а затем три каскада свертка-свертка-свертка-подвыборка. Размер узла свертки во всех слоях 3х3. При подвыборке производится выбор максимального значения (Max Pooling) из квадрата размером 2х2.
2. Вторая часть отвечает за **классификацию объекта** на изображении по выделенным на предыдущем этапе признакам. Эта часть содержит 3 полносвязных слоя. В первых двух слоях 4096 нейронов, а в последнем - 1000 нейронов (по количеству классов объектов).

Нейронная сеть VGG16 в Keras

На вход сеть VGG16, получает изображение размером 224х224 пиксела, 3 канала цвета (красный, зеленый и синий). На выходе сеть выдает вероятности (в формате one hot encoding), что на изображении тот или иной класс объекта.

Использовать сеть VGG16 в Keras, как и другие предварительно обученные сети, очень легко. Сначала нужно подключить необходимые нам модули:

from keras.applications.vgg16 import VGG16

from keras.preprocessing import image

from keras.applications.vgg16 import preprocess\_input, decode\_predictions

import numpy **as** np

Затем создаем сеть VGG16 и загружаем веса, обученные на наборе данных ImageNet:

model **=** VGG16(weights**=**'imagenet')

При первом запуске программы эта команда может выполнятся достаточно долго, т.к. необходимо загрузить большой [HDF5 файл с сохраненными весами](https://www.asozykin.ru/deep_learning/2017/02/12/How-to-save-trained-deep-net.html). Затем файл сохраняется в кэш Keras на диске и загрузка весов будет происходить быстрее.

Загружаем картинку, преобразуем ее в массив numpy и выполняем предварительную обработку:

img **=** image**.**load\_img('image\_file\_name.jpg', target\_size**=**(224, 224))

x **=** image**.**img\_to\_array(img)

x **=** np**.**expand\_dims(x, axis**=**0)

x **=** preprocess\_input(x)

Выполняем распознавание объекта на изображении:

preds **=** model**.**predict(x)

Результат распознавания - это массив из 1000 элементов. Выберем 3 элемента с самой высокой вероятностью и напечатаем их:

**print**('Результаты распознавания:', decode\_predictions(preds, top**=**3)[0])

Программа выдает следующие результаты (пример):

Результаты распознавания: [('n02124075', 'Egyptian\_cat', 0.71789128), ('n02123045', 'tabby', 0.079336673), ('n02123159', 'tiger\_cat', 0.048442803)]

Первый результат - Egyptian\_cat, его вероятность примерно 72%. Число 02124075 - это метка класса Egyptian\_cat в наборе данных ImageNet. На втором месте tabby (это тоже порода кошки) с вероятностью примерно 8%. Затем идет tiger\_cat с вероятностью меньше 5%.

Перенос обучения

Сейчас одним из самых популярных подходов к практическому обучению глубоких нейронных сетей является перенос обучения (по-английски transfer learning). В этом случае берется предварительно обученная сеть, например, VGG16 или ResNet, и дообучается на наших данных. При таком подходе данных для обучения нужно гораздо меньше, т.к. сеть уже обучена выделять нужные нам признаки, например, на наборе данных ImageNet. Времени для дообучения, как правило, тоже требуется меньше, по сравнению с обучением с нуля. При этом, возможно, придется менять архитектуру сети, чтобы она решала нужные нам задачи.

Итоги

Мы рассмотрели, как применять предварительно обученную сеть VGG16 в Keras для распознавания объектов на изображениях. Попробуйте провести свои эксперименты с сетью и оценить качество распознавания

**Также попробуйте заменить сеть VGG16 на ResNet50** (Используйте модуль Applications из документации Keras).