**Перенос обучения и расширение данных**

**Перенос обучения**

* **Преимущества использования предварительно обученных нейронных сетей:**
* Архитектуры нейронных сетей, решающих реальные задачи, часто состоят из десятков слоев и имеют миллиарды параметров, которые необходимо обучить
* Обучение нейронной сети может занимать дни и месяцы. И это обучение с одним заранее заданными гиперпараметрами
* К тому же не у всех ест достаточные ресурсы
* Перенос обучения – использование предварительно обученной сети для других задач
* Предварительно обученные нейронные сети в **Keras**
* VGG16, VGG19, inception v3, ResNet50, Xception и др.
* Набор данных для обучения ImageNet
* 1000 типов объектов

**Обзор существующих предварительно обученных нейронных сетей**

* **VGG**
* Глубокая нейронная свёрточная сеть, имеющая последовательное применение слоев и из-за этого все же ограниченная по глубине.
* Имеет две версии: 16 и 19 слоёв
* **ResNet**
* Использует непоследовательные преобразования, периодически объединяя (суммированием) выход текущего слоя с выходом слоя, который был обработан на несколько шагов ранее.
* **Inception**
* Основана на блоках с древовидной архитектурой.
* Этот блок параллельно применяет к слою несколько операций, а затем агрегирует всю информацию снова в один слой.
* **Dance Net**
* Состоит из «плотных» блоков.
* Каждый блок принимает на вход все предыдущие карты признаков.
* При этом объединение карт, в отличие от Reset, где происходит суммирование карт признаков, происходит конкатенированием
* **Exception**
* Отличается тем, что отдельно выполняет обработку и преобразование канальной информации, а отдельно -пространственной.
* Канальная информация извлекается свертками 1x1, а пространственная выделяется сверткой 3х3, но к каждому каналу по отдельности

**Перенос обучения**

* Применить на практике данные архитектуры можно, просто достав их из модуля TensorFlow

from tensorflow.keras.applications.vgg16 Import VGG16

from tensorflow.keras.applications.vgg9 Import VGG19

from tensorflow.keras.applications.resnet50 Import ResNet50

from tensorflow.keras.applications.inception\_v3 import inceptionV3

from tensorflow.keras.applications.densenet import DansaNet201

from tensorflow.keras.applications.exception import Xception

model = VGG16(weights='imagenet')

model.summary()

* Где **weights** - данные, на которых обучены веса
* В документации можно прочитать про другие архитектуры

https://www.tensorflow.org/api docs/python/tf/keras/applications

**Основные этапы переноса обучения:**

* Загрузка свёрточной части предварительно обученной нейронной сети
* Добавление к свёрточной части нового классификатора для своего набора изображений
* Заморозка веса в предварительно обученной части сети
* Обучение полученной составной сети на новом наборе данных
* Оценка качества обучения на тестовом наборе данных
* Разморозка нескольких свёрточных слоёв
* Выполнение тонкой настройки нейронной сети на новом наборе данных
* Оценка качества обучения на тестовом наборе данных