# День 5: автоэнкодеры, глубокие сети, GANs, Style Transfer.

## 1 Pretrained VGG16

Используем pretrained сеть VGG16 для классификации изображений.

1. Классификация. Разобраться с кодом в файле `day-4/solutions/visualize_results.ipynb`. Запустить на паре примеров.
2. Сегментация. Разобраться с кодом в файле `day-4/solutions/visualize_results_segm.ipynb`. Запустить на паре примеров.
3. Сегментация со скользящим окном. Закройте часть изображения на экране серым квадратом и посмотрите как изменится скор для заданного класса. Если провести это для каждого участка изображения, то получися хитмап важности данной области для данного класса.
4. ** Повторить эксперимент используя другую сеть [1]. Веса доступны по ссылке `https://github.com/tensorflow/models/tree/master/research/slim`.

## 2 Автоэнкодеры для реконструкции изображений.

1. Запустить код `autoencoder_denoising/convolutional_autoencoder_starter.ipynb'`
2. Попробовать заменить текущий лосс на `cross_entropy_loss`. Как изменились картинки?
3. Как изменится перформанс если сделать архитектуру меньше.
4. На сколько хорошо восстанавливается изображение?
5. Заменить MNIST на SVHN датасэт[2].
6. ** Попробуйте подавать на вход энкодера чб изображение из SVHN датасэта, а на выходе получать оригинальное цветное.
7. ** Предыдущее задание не будет хорошо работать, если не подавать несколько случайных цветных пикселей. Т.е. сделать почти все пиксели ч/б, кроме 20 случайных пикселей на картинке.

[1] `https://github.com/tensorflow/models/blob/master/research/inception/inception/slim/README.md`
[2] `http://ufldl.stanford.edu/housenumbers/`

## 3 GANs

1. Просмотреть и запустить ноутбук `day-5/gan/gan_notebook_starter.ipynb`. Тренировка может занять очень много времени, до 20 часов на GPU. Так что стоит останавливать раньше.
2. Если сеть тренируеся слишком медленно - уменьшить количество параметров сети.
3. Попробовать заменить MNIST на SVHN и повторить тренировку.