В этом задании вы научитесь работать с TensorFlow и тренировать генеративно-состязательные сети (Generative Adversarial Network)

**Задание:** в коде тренировки GAN пропущены имплементации некоторых функций. Имплементируйте их и запустите тренировку GAN. В начале тренировки вы увидите шумовые картинки. Потом изображения будут становиться четче, и в конце вы получите изображения похожие на:

|  |  |  |
| --- | --- | --- |
| Fully Connected GAN (MNIST) | Deep Convolutional GAN (MNIST) | Deep Convolutional GAN (CIFAR10) |
| C:\Users\dyashuni\Desktop\1.png | C:\Users\dyashuni\Desktop\2.png | C:\Users\dyashuni\Desktop\3.png |

**Пояснения к заданию**

Для генерации изображений GAN-ы используют две нейронные сети. Одна сеть – генератор ***G*** – генерирует изображения из шумового распределения. Другая сеть – дискриминатор ***D***, принимает на вход сгенерированные изображения и изображения из тренировочного датасета и пытается определить какое изображение ей пришло на вход, настоящее или сгенерированное. Цель генератора – обмануть дискриминатор, т.е. сгенерировать правдоподобные изображения, на которых дискриминатор будет ошибаться.

Процесс обучения GAN, когда генератор пытается обмануть дискриминатор, а дискриминатор пытается отличить настоящие изображения от искусственных, можно рассматривать как минимаксную игру:



изображения из тренировочного набора, шумовое распределение, из которого генератор с помощью нейронной сети генерирует изображения . ***D*** – возвращает вероятность того, что изображение настоящее.

Для нахождения решения можно последовательно обновлять веса нейронных сетей генератора и дискриминатора:

1. обновляем веса генератор ***G***, чтобы минимизировать вероятность верного ответа дискриминатора ***D***
2. обновляем веса дискриминатора ***D***, чтобы максимизировать вероятность верного ответа дискриминатора ***D***

Такая процедура обновлений весов полезна для теоретического анализа, но на практике она работает не очень хорошо, т.к. появляется проблема исчезающих градиентов для генератора, когда дискриминатор с большой уверенностью предсказывает, что изображение искусственное. На практике лучше обновлять веса генератора так, чтобы **максимизировать вероятность ошибки дискриминатора**. Этот стандартный прием позволяет улучшить ситуацию с исчезающими градиентами и применяется во многих статьях.

В этом задании мы будем следующим образом обновлять веса генератора и дискриминатора:

1. обновляем веса генератор ***G***, чтобы максимизировать вероятность ошибки дискриминатора ***D***на сгенерированных данных  
   
2. обновляем веса дискриминатора ***D***, чтобы максимизировать вероятность верного ответа дискриминатора ***D*** на реальных и сгенерированных данных  
   

В TensorFlow удобнее решать задачу минимизации, поэтому функцию потерь генератора ***G*** запишем в виде:



Функция потерь дискриминатора ***D***:



Датасеты MNIST и CIFAR10 возьмите отсюда <https://github.com/dyashuni/HSE_2017/tree/master/Datasets>  
и обновите mnist\_path и cifar10\_path переменные в коде.

**Бонус**: сделайте что-нибудь дополнительное и получите бонусные баллы.

**Форма сдачи**: устно, покажите ваши имплементации функций и сгенерированные изображения. Чем отличаются изображения, сгенерированные полно-связными (fully connected) и сверточными (convolutional) нейронными сетями и почему?