CycleGAN

Автор доклада: Ярослав Спирин

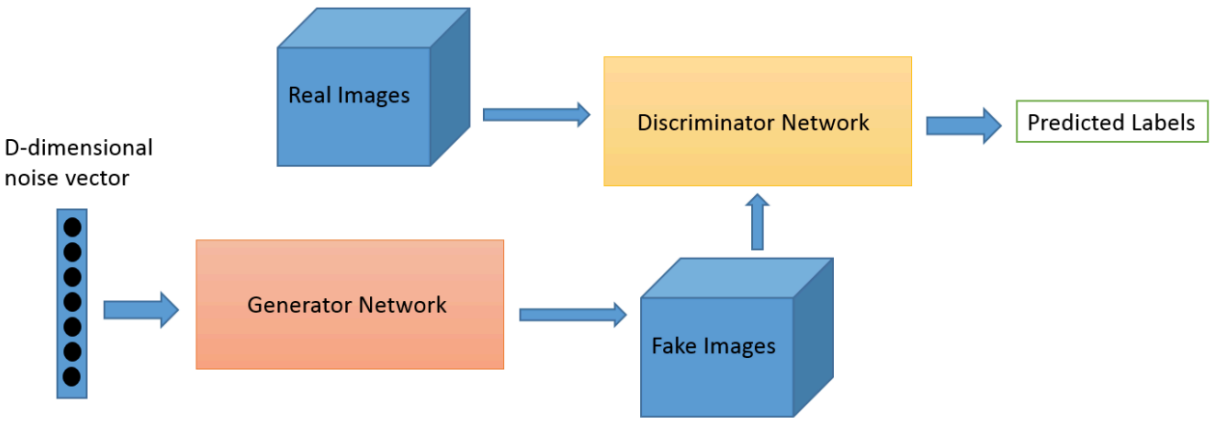
Original: Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks

Введение

В докладе будет рассмотрена архитектура нейронной сети, называемая CycleGAN *Cycle Generative Adversarial Network.* Подобное семейство алгоритмов принадлежит к множеству генеративных моделей, которые в свою очередь относятся к области машинного обучения называемой ***unsupervised learning*** (обучение без учителя, то есть без указания target переменной).

Генеративно-Состязательные сети состоят из двух основополагающих моделей (частей):

- Первая модель называется Генератор, и ее цель заключается в том, чтобы генерировать новые объемы данных, похожими на ожидаемые. Генератор можно рассматривать как человека, который рисует подделки на реальные произведения искусства великих мастеров.
- Вторая модель —это Дискриминатор. Задача этой модели заключается в том, чтобы распознавать, что данные, переданные ей на вход — это “настоящая”



картина или “подделка” (т.е. принадлежит оригинальному датасету или является копией, которая была сгенерирована Генератором)

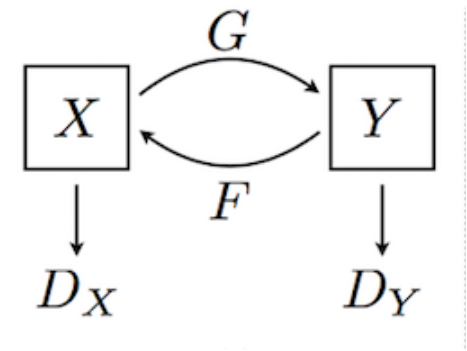
При этом функции потерь для обеих моделей выглядят следующим образом:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log \left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

Формула градиента для генератора, данное слагаемое максимизирует вероятность сгенерированных данных G(z).

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log \left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right)\right].$$

Формула градиента для дискриминатора, первое слагаемое максимизирует вероятность реальных данных D(z), а второе слагаемое минимизирует вероятность сгенерированных данных G(z).



Наивная архитектура CycleGAN

Архитектура CycleGAN

Архитектура состоит из двух частей:

- Два отображения G: X ->Y и F: Y -> X
- Соответствующие состязательные дискриминатора Dx и Dy

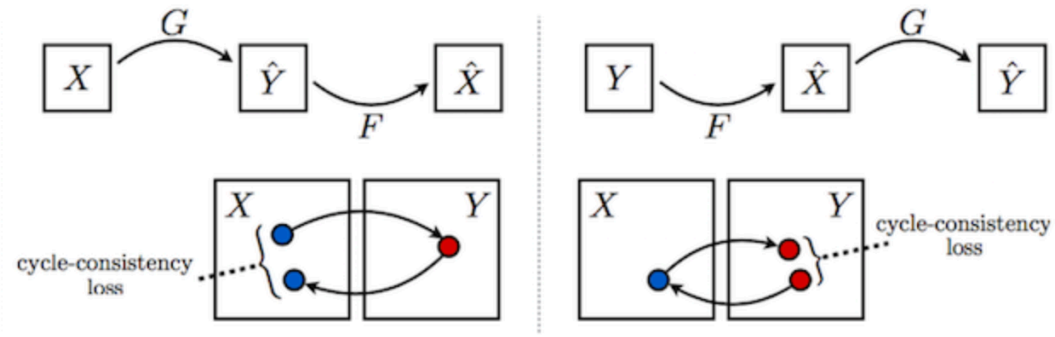
Задача отображения G заключается здесь в том, чтобы перевести множество X во множество образов, которые подаются на вход дискриминатору Dy для

определения того, что образы являются оригинальными или подделками в соответствии с доменом Dy.

Задача отображения F заключается здесь в том, чтобы перевести множество Y во множество образов, которые подаются на вход дискриминатору Dx для определения того, что образы являются неотличимыми от домена Dx.

Функция потерь

Метод берет свое название именно из-за функции потерь, который помимо стандартных для всех архитектур типа GAN функций потерь генератора и дискриминатора еще используют *цикличную функцию потерь.*



Логика, которая стоит за данным называем следующая: функция потерь определяется как разница между тем, что получается при переводе X в Y преобразованием G и последующим преобразованием F полученного образа обратно в X. Такая функция потерь должна быть минимизирована.

Итоговая формула для функции потерь:

$$\mathcal{L}(G, F, D_X, D_Y) = \mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) + \mathcal{L}_{\text{GAN}}(F, D_X, Y, X) + \lambda \mathcal{L}_{\text{cyc}}(G, F),$$

Результат

Как результат работы данной архитектуры нейронной сети можно видеть массу развлекательных мобильных приложений в магазинах приложений App Store и Google Play. Одно из них мне даже посчастливилось написать собственноручно: https://apps.apple.com/ru/app/lambda-app/id1448239406