face2comic

Гиркин Валерий Стрельцов Антон

Аннотация

В рамках данного проекта мы планируем реализовать перевод фотографий человеческих лиц в изображения персонажей аниме при помощи CycleGAN. Нас заинтересовала архитектура нейросети CycleGAN, в основном своими результатами и тем, что для нее не обязательно размечать данные.

1 Введение

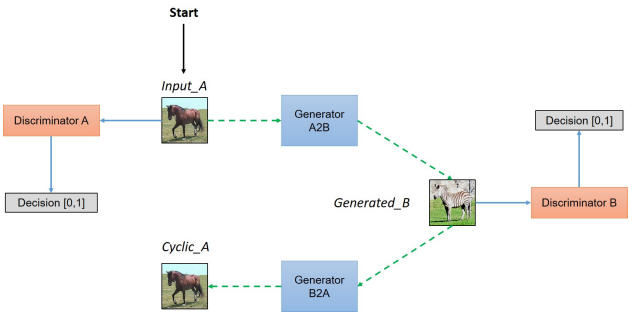
Постановка задачи следующая – есть 2 датасета: фотографии реальных людей и картинки аниме персонажей, нужно обучить модель переводить их друг в друга. Так как зависимостей между ними нет, то задача является задачей обучения без учителя. Потому было решено взять за основу CycleGAN. Именно такая задача еще не решалась, хотя существуют решения генерации отдельно реальных лиц и отдельно аниме на основе DCGAN, также мы нашли работу, в которой лица переводились в мультяшные лица с помощью придуманной ими XGAN, но эта работа была очень емкой: помимо множества лоссов также перебиралось много параметров, из-за чего воспризводимость работы довольна низкая, что было отмечено в ревью на openreview.net. Тем не менее одну идею мы оттуда задействовали.

2 Данные

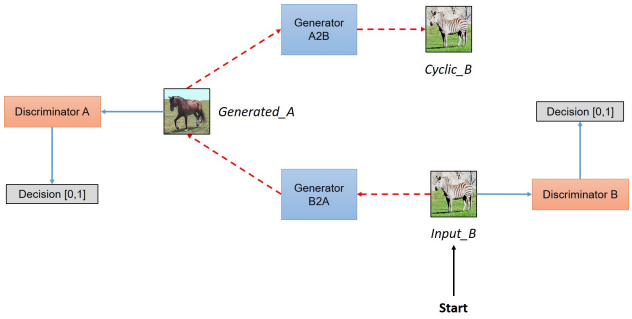
Нам нужны были фотографии лиц реальных людей и персонажей аниме. Готовых размеченых датасетов мы не нашли, поэтому решили использовать неразмеченные данные. В качестве выборки аниме был скачан датасет Danbooru Faces, скачивать данные самим предстваляло достаточно нетривиальную задачу, т.к. многие сайты заблокированы на территории РФ. Большая проблема в том, что на сайтах-галлереях на подавляющем количестве картинок изображены девушки, потому фотографии нам пришлось также взять в основном с лицами женщин. Изначально был выбран "IMDB-WIKI – 500k+ face images with age and gender labels во-первых. из-за его объема, а во-вторых, из-за его возрастной и половой разметки. Но он оказался не совсем удачным: датасет использовался для детектирования лиц, а не их распознования, потому на некоторых картинках лица были очень мелкими, а картинки дополнены краями изображения, что мешало обучению. В итоге был выбран датасет CelebsA. Были отброшены мужские фотографии. В итоге осталось 120к фотографий, 120к из примерно 440к картинок аниме были выбраны случайно.

3 Архитектура

Простейший CycleGAN устроен следующим образом.



Две пары генератор-дискриминатор соответствующие двум выборкам. На каждом шаге обучения по очереди происходит обучение в обе стороны: сначала на вход генератору А и дискриминатору А подается картинка из выборки А, выход генератора подается на вход дискриминатору Б и генератору Б, считается лосс дискриминаторов. Дальше считается лосс генератора А, по тому, насколько похожая на вход в получилась картинка у генератора Б, а также ошибке дискриминатора Б, после чего последовательность повторяется для выборки Б. Хорошо обученные CycleGAN'ы



часто используют дополнительные бинарные метки классов, например, есть ли какой-то определенный объекты, или обладает ли объект на картинке определенным признаком. Тогда между некоторыми объектами выборок А и Б возникает соответствие. В нашей работе такого, к сожалению, не будет, т.к. в картинках аниме совсем нет никакой разметки.
Также мы поэкспериментировали с этой архитектурой. Например, увеличивали размер ядер конволюций, заменяли дискриминаторы на критиков, предложенных в статье о WGAN, и добавляли дискриминатор на представления после энкодеров в генераторах, который должен способствовать приведению эмбеддингов в одно пространство.
Лучшие результаты показали обычный CycleGAN с увеличенными ядрами и его улучшение с дискриминатором между эмбеддингами.

X_1 - аниме, G_{12} - генератор из аниме в реальные, G_{emb12} - активации среднего слоя генератора из аниме в реальные (ембеддинги), D_2 - дискриминатор, отличающий фейковый реальные картинки от оригинальных реальных. В обратную сторону индексы меняются. Также есть D_{emb} - дискриминатор отличающий эмбеддинги двух генераторов друг от друга. В итоговой версии лоссы следующие:

$$\begin{aligned} L_{generator12} \\ = \frac{1}{batch\_size}\sum\left((D_2(G_{12}(X_2))-1)^2\right) \\ + 0.5\frac{1}{batch\_size}\sum\left((D_{emb}(G_{emb12}))^2\right) \\ + \frac{1}{batch\_size}\sum\left((X_1 - G_{21}(G_{12}(X_1)))\right) \end{aligned} \quad (1)$$

Превый лосс в сумме – лосс обмана дискриминатора, второй – лосс обмана эмбеддингового дискриминатора, последний – лосс реконструкции.

$$\begin{aligned} L_{discriminator2} = \frac{1}{batch\_size}\sum\left((D_2(G_{12}(X_2)) \right. \\ \left. - uniform\_noise(0,0.2))^2\right) \end{aligned} \quad (2)$$

Добавление шума хорошо сказалось на итоговом качестве картинок.
Для лоссов цикла в другую сторону просто меняются индексы.

$$\begin{aligned} L_{emb\_dicriminator} = \frac{1}{batch\_size}\sum(D_{emb}(G_{emb12}))^2 \\ + \frac{1}{batch\_size}\sum(D_{emb}(G_{emb21}) \\ - 1)^2 \end{aligned} \quad (3)$$

4 Результаты

Рис. 1: Пример преобразования аниме лиц в реальные, в нечетных столбцах – изображения на входе, в четных – соответствующие выходы.

Приведенные примеры – результат 55к итераций (заняло примерно 4 часа) обычного CycleGAN. Как можно увидеть, сеть действительно обучается, но недостаточно хорошо. И на самом деле очень приятно удивил факт того, что цвет волос, положение головы и выражение лица часто совпадают. Основная проблема в том, что нет никаких связей между двумя выборками.

Результаты CycleGAN с дискриминатором на эмбеддингах. Визуально лучше обыного.

Результаты WassersteinCycleGAN оказались гораздо хуже. Параметры были взяты из оригинальной статьи по WGAN:

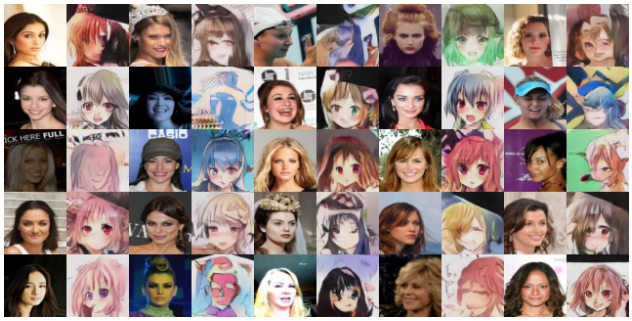


Рис. 2: Пример преобразования реальных лиц в аниме, в нечетных столбцах – изображения на входе, в четных – соответствующие выходы.

Рис. 3: Пример преобразования аниме лиц в реальные, в нечетных столбцах – изображения на входе, в четных – соответствующие выходы.

5 Выводы

Ни один из способов не дал безупречных результатов, но полученные результаты вполне удовлетворительны и наверняка могли бы быть улучшены с помощью тьюнинга параметров. Но для этого потребовалось бы больше времени и вычислительных ресурсов.

6 Ссылки

Датасет аниме лиц –
https://gist.github.com/stormraiser/
a8066517b0b60a50c701ee9c8f720691
Датасет лиц –
https://data.vision.ee.ethz.ch/cvl/rrothe/imdb-wiki/
Пример CycleGAN –
https://github.com/yunjey/mnist-svhn-transfer

Рис. 4: Пример преобразования реальных лиц в аниме, в нечетных столбцах – изображения на входе, в четных – соответствующие выходы.

Рис. 5: Пример преобразования аниме лиц в реальные, в нечетных столбцах – изображения на входе, в четных – соответствующие выходы.

Хаки для GAN –
https://github.com/soumith/ganhacks
XGAN –
https://arxiv.org/abs/1711.05139
WGAN –
https://arxiv.org/pdf/1701.07875

Рис. 6: Пример преобразования реальных лиц в аниме, в нечетных столбцах – изображения на входе, в четных – соответствующие выходы.