Advanced Computer Vision

Практический курс

Савельева Юлия Олеговна, 3 семестр, 02.11.2021

Wasserstein GAN

Данные

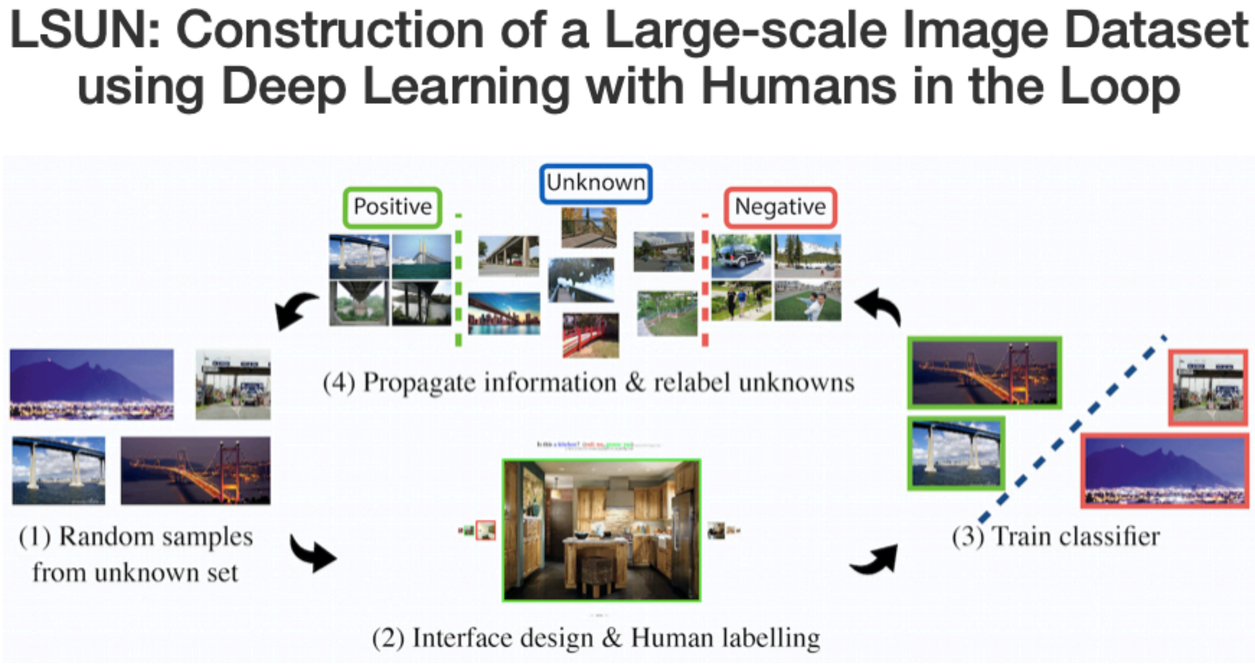
Скачать датасеты MNIST и LSUN.
Оба датасета доступны в torch Datasets. MNIST можно скачать, указав соответствующий параметр в конструкторе, но LSUN нужно скачать заранее и передать путь до датасета в конструктор:

https://github.com/fyu/lsun

Команда для скачивания:
python download.py -o path/to/save/lsun/dataset/on/your/machine -c bedroom

Создание датасета в коде:

dataset = torch vision.datasets.LSUN(root='path/to/save/lsun/dataset/on/your/machine', classes=['bedroom_train'], transforms=preprocessing_transforms)



Wasserstein GAN

Архитектура, инициализация и препроцессинг

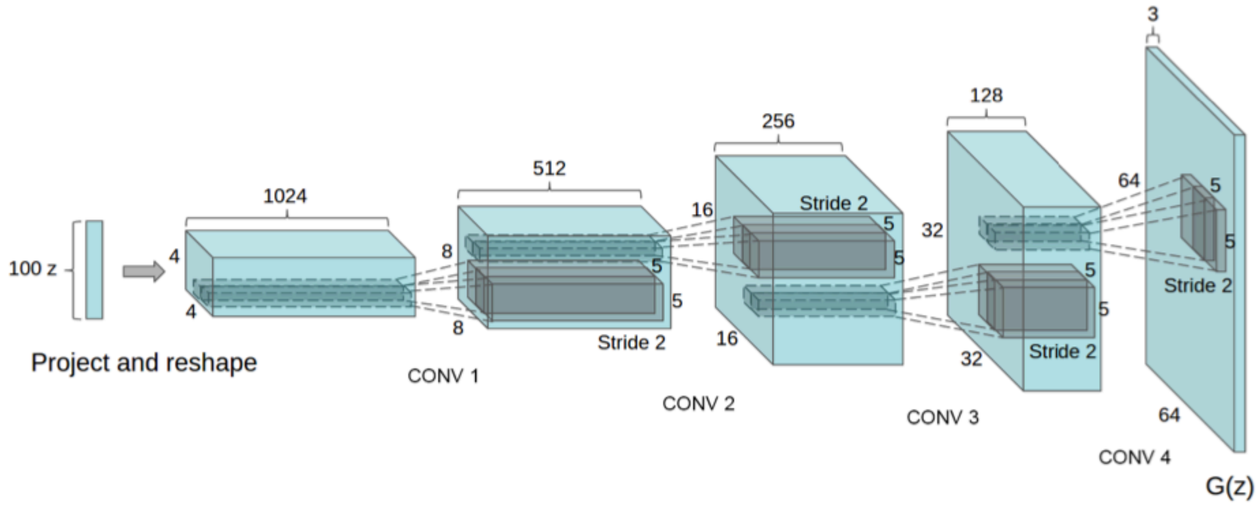
Реализовать нейронную сеть из статьи DCGAN:

https://arxiv.org/pdf/1511.06434.pdf

Здесь можно посмотреть реализацию архитектуры, инициализации весов и препроцессинга (но избавиться от Sigmoid в дискриминаторе!):

https://pytorch.org/tutorials/beginner/dcgan_faces_tutorial.html

P.S. В датасете MNIST картинки 28x28, просто применяем к ним такой же препроцессинг, чтобы они растянулись до 64x64.



Wasserstein GAN

Целевая функция

Статья, на которую опираться при реализации:
https://arxiv.org/pdf/1704.00028.pdf

Гиперпараметры из псевдокода оставить такими же.

Реализация подсчета gradient penalty находится в функции TOGradOfDisc:
https://github.com/jalola/compare-tensorflow-pytorch/blob/master/test/gan_pytorch_model.py

Algorithm 1 WGAN with gradient penalty. We use default values of $\lambda = 10$, $n_{\text{critic}} = 5$, $\alpha = 0.0001$, $\beta_1 = 0$, $\beta_2 = 0.9$.

Require: The gradient penalty coefficient λ, the number of critic iterations per generator iteration n_{critic}, the batch size m, Adam hyperparameters α, β_1, β_2.
Require: initial critic parameters w_0, initial generator parameters θ_0.

while θ has not converged **do**
  for $t = 1, ..., n_{\text{critic}}$ **do**
    for $i = 1, ..., m$ **do**
      Sample real data $\boldsymbol{x} \sim \mathbb{P}_r$, latent variable $\boldsymbol{z} \sim p(\boldsymbol{z})$, a random number $\epsilon \sim U[0, 1]$.
      $\tilde{\boldsymbol{x}} \leftarrow G_\theta(\boldsymbol{z})$
      $\hat{\boldsymbol{x}} \leftarrow \epsilon \boldsymbol{x} + (1 - \epsilon)\tilde{\boldsymbol{x}}$
      $L^{(i)} \leftarrow D_w(\tilde{\boldsymbol{x}}) - D_w(\boldsymbol{x}) + \lambda(\|\nabla_{\hat{\boldsymbol{x}}} D_w(\hat{\boldsymbol{x}})\|_2 - 1)^2$
    end for
    $w \leftarrow \text{Adam}(\nabla_w \frac{1}{m} \sum_{i=1}^{m} L^{(i)}, w, \alpha, \beta_1, \beta_2)$
  end for
  Sample a batch of latent variables $\{\boldsymbol{z}^{(i)}\}_{i=1}^{m} \sim p(\boldsymbol{z})$.
  $\theta \leftarrow \text{Adam}(\nabla_\theta \frac{1}{m} \sum_{i=1}^{m} -D_w(G_\theta(\boldsymbol{z})), \theta, \alpha, \beta_1, \beta_2)$
end while

Wasserstein GAN

Целевая функция

Статья, на которую опираться при реализации:
https://arxiv.org/pdf/1704.00028.pdf

Гиперпараметры из псевдокода оставить такими же.

Реализация подсчета gradient penalty можно подсмотреть в функции TOGradOfDisc:
https://github.com/jalola/compare-tensorflow-pytorch/blob/master/test/gan_pytorch_model.py , где

interpolates = eps * real + (1 - eps) * fake

Algorithm 1 WGAN with gradient penalty. We use default values of $\lambda = 10$, $n_{\text{critic}} = 5$, $\alpha = 0.0001$, $\beta_1 = 0$, $\beta_2 = 0.9$.

Require: The gradient penalty coefficient λ, the number of critic iterations per generator iteration n_{critic}, the batch size m, Adam hyperparameters α, β_1, β_2.
Require: initial critic parameters w_0, initial generator parameters θ_0.

while θ has not converged **do**
  for $t = 1, ..., n_{\text{critic}}$ **do**
    for $i = 1, ..., m$ **do**
      Sample real data $\boldsymbol{x} \sim \mathbb{P}_r$, latent variable $\boldsymbol{z} \sim p(\boldsymbol{z})$, a random number $\epsilon \sim U[0, 1]$.
      $\tilde{\boldsymbol{x}} \leftarrow G_\theta(\boldsymbol{z})$
      $\hat{\boldsymbol{x}} \leftarrow \epsilon \boldsymbol{x} + (1 - \epsilon)\tilde{\boldsymbol{x}}$
      $L^{(i)} \leftarrow D_w(\tilde{\boldsymbol{x}}) - D_w(\boldsymbol{x}) + \lambda(\|\nabla_{\hat{\boldsymbol{x}}} D_w(\hat{\boldsymbol{x}})\|_2 - 1)^2$
    end for
    $w \leftarrow \text{Adam}(\nabla_w \frac{1}{m} \sum_{i=1}^{m} L^{(i)}, w, \alpha, \beta_1, \beta_2)$
  end for
  Sample a batch of latent variables $\{\boldsymbol{z}^{(i)}\}_{i=1}^{m} \sim p(\boldsymbol{z})$.
  $\theta \leftarrow \text{Adam}(\nabla_\theta \frac{1}{m} \sum_{i=1}^{m} -D_w(G_\theta(\boldsymbol{z})), \theta, \alpha, \beta_1, \beta_2)$
end while

Wasserstein GAN

Процесс обучения

При обучении советую мониторить значение целевой функции генератора и дискриминатора, а также точность дискриминатора на реальных и сгенерированных картинках.

Обучение лучше начать с генерации только одной буквы из MNIST!

Этот тьюториал полезен для того, чтобы понимать, как процесс обучения будет идти и сколько ждать результатов (архитектуру отсюда не использовать):
https://machinelearningmastery.com/practical-guide-to-gan-failure-modes/

Дедлайн 16.11.2020 00:00