Do deep generative...

Сумекенов Ахмад, Жилкина Ксения

Декабрь 2019

Аннотация

В этом отчете мы рассмотрим и обсудим статью "Do Deep generative models know that they don't know"[1]. Очень много работы и примеров в сфере глубинного обучения посвящено тому, что, классические дискриминаторы очень легко обмануть, давая немного измененные картинки, либо картинки из другого домена, что при высокой уверенности они дают неправильные результаты. Обычно думается, что генеративные модели менее склонны к таким вещам, однако авторы стать показывают, что это не так, на примере моделей PixelCNN, VAE и Glow, показывая что они например дают более высокую вероятность сэмплу из SVHN(это адресные номера, висящие на домах), чем из собственного домена(тренировали они на CIFAR-10).

1 Введение

Мы знаем, что глубинное обучение достигло больших высот в классификации и моделирования условного предсказания $p(y|x)$, например, если взять тот же ImageNet, то уже несколько лет как созданы модели, которые предсказывают метку, лучше чем человек. Тем не менее, хорошо создавая модели на данных из своего домена, они очень плохо работают на данных не из распределения. Одним из способов решить эту проблему сделать модель, которая будет предсказывать не класс, или что-то другое из объекта, а вероятность того, *что этот объект принадлежит нашему распределению.*

Зачем это нужно? Дело в том, что это не только решит проблему "обманывания"моделей и "атак"на них, но также будет использоваться в других сферах. Например в anomaly detection и uncertainty estimation.

2 Модели

Попробуем обучить Gated PixelCNN и VAE на MNIST и посмотрим как они генерируют цифры.

Для VAE мы использовали следующую архитектуру.

Посмотрим на картинки, генерируемые на разных эпохах:

```python
class VAE(nn.Module):
    def __init__(self):
        super(VAE, self).__init__()

        self.fc1 = nn.Linear(784, 400)
        self.fc21 = nn.Linear(400, 20)
        self.fc22 = nn.Linear(400, 20)
        self.fc3 = nn.Linear(20, 400)
        self.fc4 = nn.Linear(400, 784)

    def encode(self, x):
        h1 = F.relu(self.fc1(x))
        return self.fc21(h1), self.fc22(h1)

    def reparameterize(self, mu, logvar):
        std = torch.exp(0.5*logvar)
        eps = torch.randn_like(std)
        return mu + eps*std

    def decode(self, z):
        h3 = F.relu(self.fc3(z))
        return torch.sigmoid(self.fc4(h3))

    def forward(self, x):
        mu, logvar = self.encode(x.view(-1, 784))
        z = self.reparameterize(mu, logvar)
        return self.decode(z), mu, logvar
```

Рис. 1: The VAE

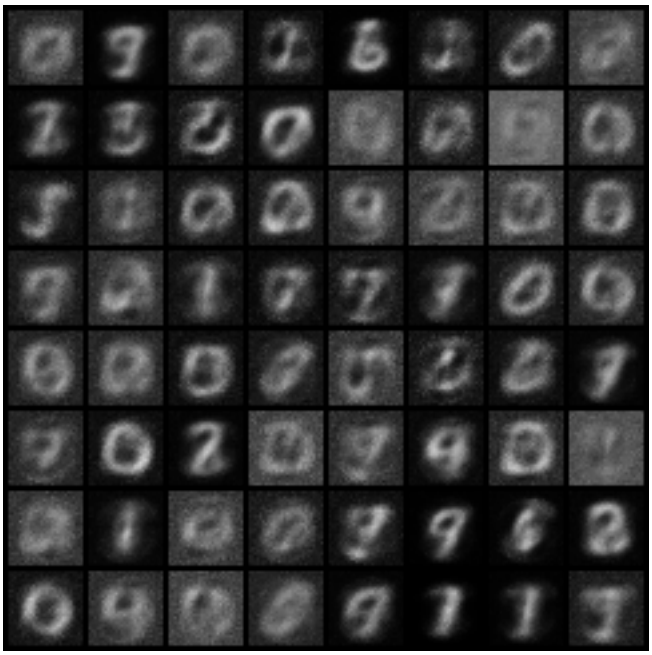

Рис. 2: Sample from 1st epoch

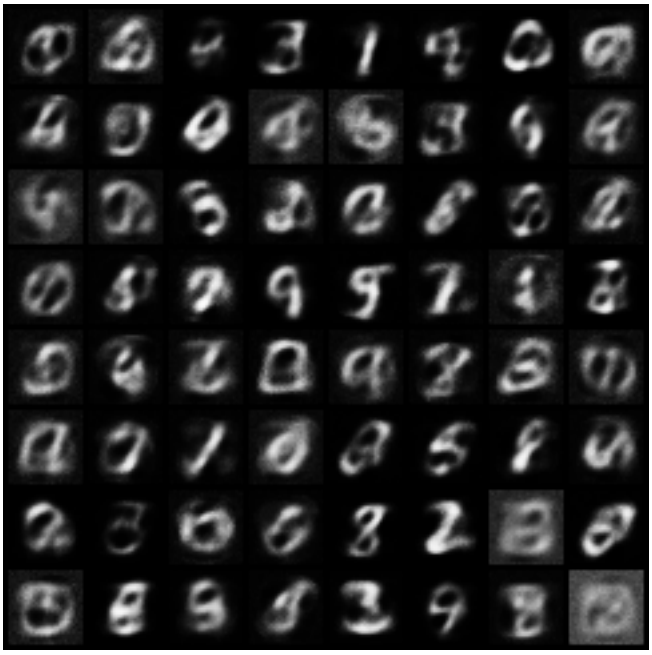

Рис. 3: Sample from 3st epoch

Рис. 4: Sample from 9st epoch

3 PixelCNN

То есть это довольно примитивный авто-энкодер, с полносвязными слоями и самым узким слоем размером в 20 нейронов.

Посмотрим на тот же MNIST, только со стороны PixelCNN:(смотрим в конце файла)

Также приложим лосс PixelCNN от MNIST и печальные результаты PixelCNN по CIFAR (а также график лосса PixelCNN по CIFAR)

4 Заключение

"I always thought something was fundamentally wrong with the universe"

(Я не забыл это убрать из теховского шаблона, я с этим согласен)

Список литературы

[1] Yee Whye Teh Dilan Gorur Balaji Lakshminarayanan Eric Nalisnick, Akihiro Matsukawa. *Do Deep Generative Models Know What They Dont Know?* ICLR, 2019.

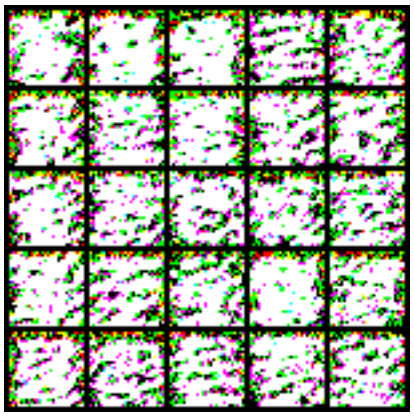

Рис. 5: Sample from 1st epoch : Gated PixelCNN

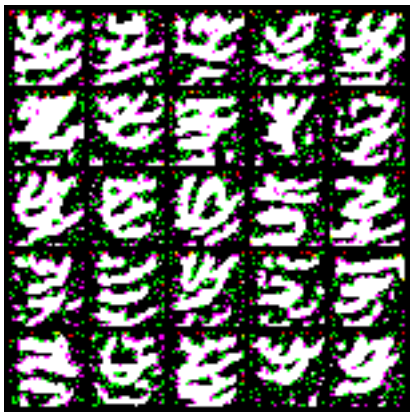

Рис. 6: Sample from 6st epoch : Gated PixelCNN

Рис. 7: Sample from 15st epoch : Gated PixelCNN

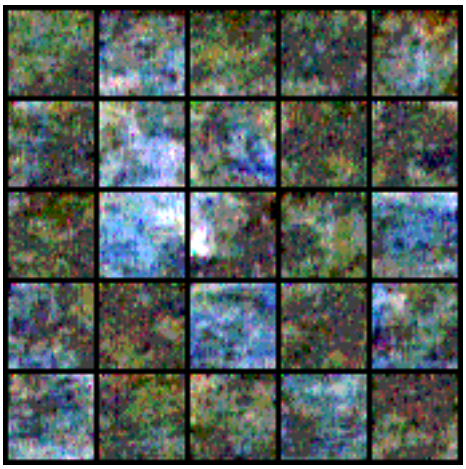

Рис. 8: PixelCNN CIFAR 1st epoch

Рис. 9: PixelCNN CIFAR 6st epoch

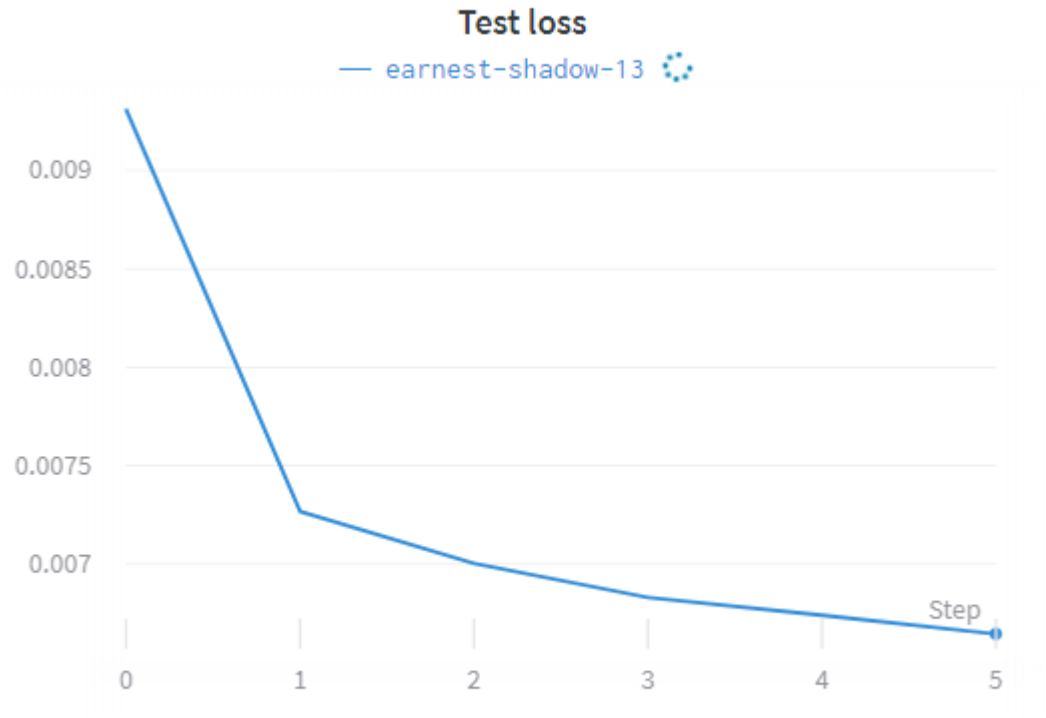

Рис. 10: PixelCNN CIFAR Loss

Рис. 11: PixelCNN MNIST Loss