Semi-supervised Learning with Deep Generative Models (воспроизведение результатов)

Даниил Гайдамашко
БПМИ-161
НИУ ВШЭ

Денис Сурин
БПМИ-162
НИУ ВШЭ

Abstract

В статье авторы пытались решить задачу частичного обучения с использованием генеративных моделей. Своей целью они ставили при помощи небольшого размеченного датасета уметь получать результаты для намного большего количества неразмеченных данных. К примеру, из датасета MNIST брали 50000(?) неразмеченных объектов и от 100 до 3000 размеченных, и на них запускали свой алгоритм (то есть объем неразмеченных мог на несколько порядков быть меньше размеченных). Непосредственно в статье авторы пробовали такой подход на задаче классификации(определение верной цифры по картинке).

1 Описание задачи и используемых алгоритмов

В качестве генеративных моделей использовался вариационный автоэнкодер с различными модификациями.

Формально, поставлена следующая задача: Наблюдаемые данные вида (X, Y), где $x \in \mathbb{R}^D$, $y \in \{1, \dots, L\}$ сопоставляются некоторым латентным переменным z. В условиях частичного обучения метки известны только у некоторых объектов выборки. Предположим, что величины X генерируются из распределения $p_\theta(x|z)$.

Чтобы найти по заданной выборке апостериорное распределение скрытых переменных $p_\theta(z|x)$, аппроксимируем его распределением $q_\phi(z|x)$.

Были предложены 3 модели: M1, M2 и M1 + M2.

1.1 M1

В качестве априорного распределения латентных переменных берется $p(z) = \mathcal{N}(z|0, I)$. Декодер строит распределение $p_\theta(x|z) = f(x; z, \theta)$, а энкодер аппроксимирует апостериорное при помощи $q_\phi(z|x) = \mathcal{N}(z|\mu_\phi(x), diag(\sigma_\phi(x)))$. Функция потерь определим при помощи нижней оценки правдоподобия:

$$\log p_\theta(x) \geq \mathbb{E}_{q_\phi(z|x)} \left[\log p_\theta(x|z)\right] - KL(q_\phi(z|x)||p_\theta(z)) = -\mathcal{J}(x)$$

где первая часть является reconstruction loss (считается при помощи reparametrization trick), а KL-loss выводится аналитически.

В M1 на вход энкодеру подавались как размеченные, так и неразмеченные данные. Энкодер представлял собой нейросеть с 2 скрытыми слоями (размерность 600) и слоем активации softplus. Далее мы получаем вектор средних mu и сигма - параметры нормального распределения, из которого семплируем вектор латентных переменных z. Z передается декодеру и по нему получаем конечный результат. После обучения автоэнкодера происходит обучение классификатора: сначала исходная картинка кодируется

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

обученным энкодером и далее на ее результатах обучается классификатор (применяется только на размеченных данных).

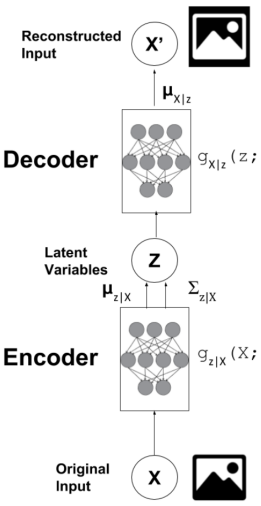

Рис. 1: M1 архитектура

1.2 M2

Для построения распределения латентных переменных энкодером используются также и метки y. Вероятностная модель следующая:

$$p(y) = Cat(y|\pi),\ p(z) = \mathcal{N}(z|0, I),\ p(x|y, z) = f_\theta(x, y, z, \theta)$$

Если говорить менее формально, M2 в автоэнкодере идет разделение: размеченные данные обучаются на классификаторе, неразмеченные - в энкодере - и латентные переменные z получаются из распределения, зависящего и от x и от y. Итоговое предсказание получается при помощи обучившегося классификатора.

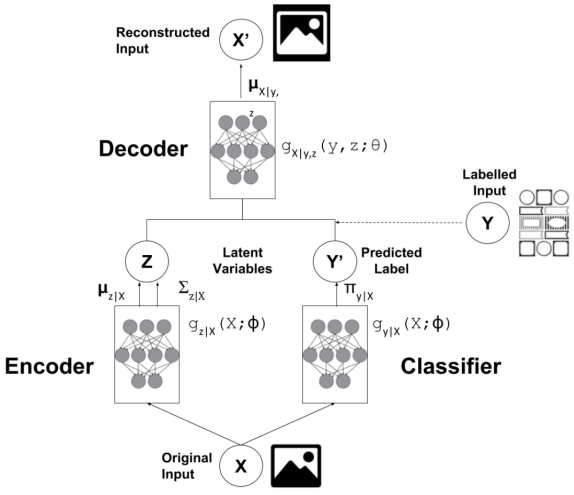

Рис. 2: M2 архитектура

1.3 M1 + M2

Данная архитектура объединяет предыдущие подходы: на вход модели M2 подается новая латентная переменная z_1, полученная в результате обучения модели M1.

$$p(x, y, z_1, z_2) = p(y)p(z_2)p_\theta(z_1|y, z_2)p_\theta(x|z_1).$$

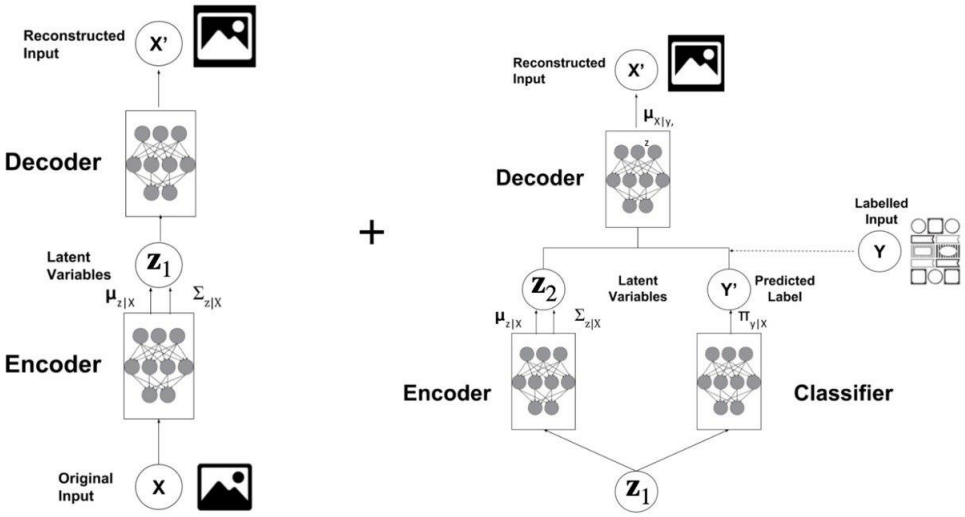

Рис. 3: M1+M2 архитектура

2 Данные

Авторы проводили эксперименты на датасетах MNIST, SVNH, NORB. Нашей текущей задачей было провести эксперименты только на датасете MNIST моделей M1, M2. Данные были нормализованы и разбиты на тестовую и обучающую выборки стандартными средствами Pytorch.

3 Подзадачи

На текущей момент стояло 2 подазадачи: реализовать модели M1 и M2, протестировать их на датасете MNIST. Модели были реализованы, в качестве гиперпараметров были взяты значения, указанные авторами статьи. Веса обеих моделей так же были проинициализированы как указано в статье. В качестве нормирующей костанты в Reconstruction loss было взято значение C = 0.5. В результатах приведены данные при N - число размеченных объектов - = 3000. Мы пробовали проводить тестирование при других значениях N, значительно отличающихся результатов получено не было.

4 Результаты

Полученные результаты пока что отличаются от авторских. Это можно объяснить тем, что в статье было довольно немного деталей реализации и если основная архитектура была бегло указано, как и некоторые гиперпараметры, то функция потерь была указана лишь аналитически, поэтому детали реализации могут значительно отличаться от авторских, что приводит к более худшим в сравнении с авторскими (у них лосс был порядка 11) результатам.

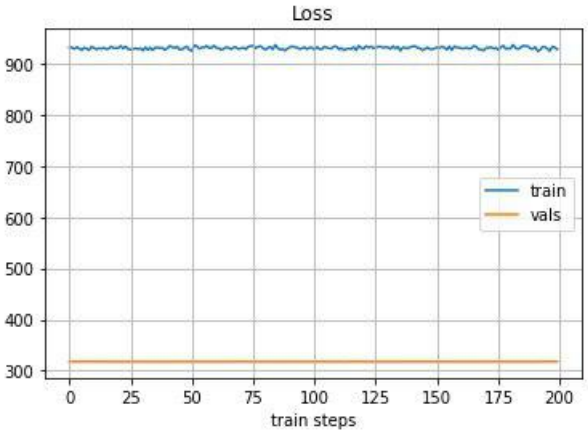

Рис. 4: M1 кривая обучения (accuracy≈ 0.1)

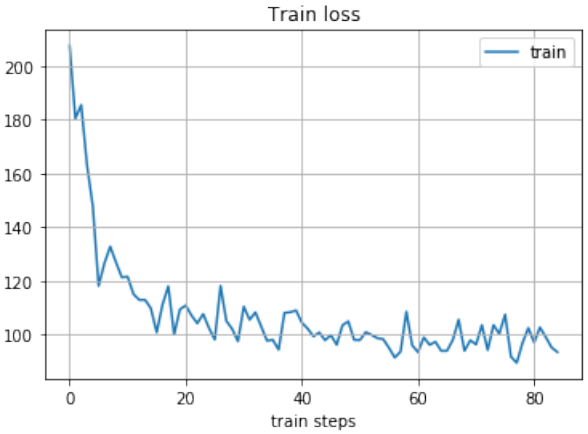

Рис. 5: M2 кривая обучения на обучающей выборке

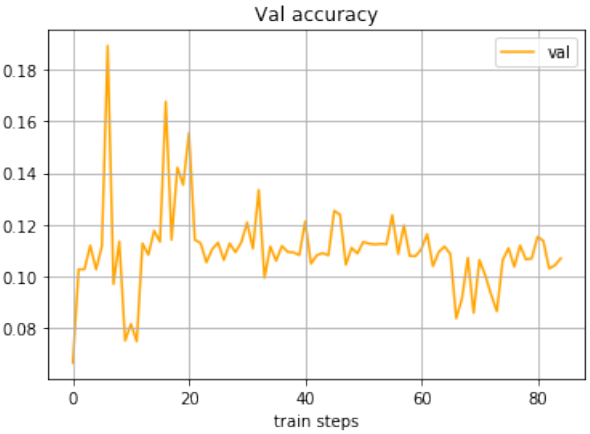

Рис. 6: M2 метрика accuracy на валидационной выборке

5 Дальнейшие планы

Следующим заданием является реализация M1 + M2 модели и тестирование на 2 датасетах - MNIST и SVHN. Так же после получения обратной связи надеемся суметь улучшить существующие модели.

6 Ссылка на репозиторий

`https://github.com/Haidaansko/SSL-VAE`