Практическая работа № 8

“Генеративные модели и автоэнкодеры”

Цель: изучить арихтектуры автокодировщиков.

Порождающие модели

Порождающие модели (англ. generative model) — это класс моделей, которые обучают совместное распределение данных p(x,y); отсюда легко получить условное распределение p(y∣x)=p(x,y)p(x), но совместное даёт больше информации и его можно использовать, например, для генерации новых фотографий животных, которые выглядят как настоящие животные.

С другой стороны, дискриминативная модель обучает только условное распределение и может, например, отличить собаку от кошки.

Примером простейшей порождающей модели является наивный байесовский классификатор.

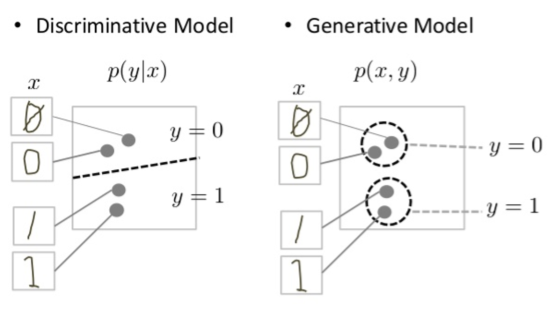


Рисунок 1. Порождающая модель пытается генерировать рукописные 0 и 1, для этого моделирует распределение по всему пространству данных. Напротив, дискриминативная модель старается разделить данные, без необходимости точно моделировать, как объекты размещаются по обе стороны от линии

Можно использовать некоторые эмпирические правила для генерации новых объектов, не используя машинного обучения.

Мы хотим научиться создавать правдоподобный объект относительно некоторой скрытой структуры исходных объектов. Давайте изучим распределение по ним, а затем просто будем сэмплировать новый объект из этого распределения. Значит эта задача относится к классу задач обучения без учителя.

Порождающая модель иногда позволяет использовать обучение с частичным привлечением учителя. Пусть задача состоит в том, чтобы отличить кошек от собак на фотографиях. Обычно мало размеченных данных, на которых кошки и со­баки отмечены вручную. Основная часть задачи со­стоит в том, чтобы понять, чем разумные фотографии отличаются от слу­чайного шума. Иначе говоря, если сначала определить распределение p(x), то проще обучить распределение p(y∣x), где y — это один бит, отвечающий за отдельный признак, а x — это вся фотография.

Таксономия порождающих моделей

Генеративные модели различаются как раз тем, как именно они строят рас­пределение p(x;θ).

Явный подход

Можно строить это распределение явно, делая вероятностные предположения, которые обычно сводятся к тому, что общее распределение p(x;θ) выражается в виде произведения тех или иных распределений.

Как правило, модели, где плотность известна явно, делают какие-то дополни­тельные предположения на структуру этих распределений.

Например, байесовские сети строят распределение из условных распре­делений

Можно даже и вовсе никаких предположений не делать: любое распределение всегда раскладывается как:

Неявный подход

Основная альтернатива всему этому состоит в том, чтобы использовать *неявные* порождающие модели, в которых мы не пытаемся получить функцию, подсчитывающую плотность нужного распределения в каждой точке, а просто мо­делируем то, что нам от этой модели нужно. Например, если мы хотим просто научиться порождать фото­графии милых котиков, нам не так важно иметь явную функцию плотности p(x)p(x), которая могла бы сказать, насколько вероятно, что перед нами котик, - вполне до­статочно просто уметь генерировать новые x∼p(x)x∼p(x).

Сэмплирование из сложных многомерных распределений делается с помощью МСМС-методов: попробуем построить мар­ковскую цепь, которая описывает случайное блуждание под графиком плотности распределения. Если достаточно долго блуждать под графиком плотности p(x)p(x), можно будет считать, что полученная точка представляет собой случайную точ­ку, взятую по распределению p(x)p(x). Примером такого моделирования глубокой сетью являются порождающие стохастические сети.

Порождающие состязательные сети — алгоритм машинного обучения, построенный на комбинации из двух нейронных сетей: генеративная модель GG, которая строит приближение распределения данных, и дискриминативная модель DD, оценивающая вероятность, что образец пришел из тренировочных данных, а не сгенерированных моделью GG. Обучение для модели GG заключается в максимизации вероятности ошибки дискриминатора DD.

Автокодировщик

Автокодировщик — специальная архитектура искусственных нейронных сетей, позволяющая применять обучение без учителя при использовании метода с обратного распространения ошибки. Простейшая архитектура автокодировщика — сеть прямого распространения, без обратных связей, наиболее схожая с перцептроном и содержащая входной слой, промежуточный слой и выходной слой. В отличие от перцептрона, выходной слой автокодировщика должен содержать столько же нейронов, сколько и входной слой.

Автокодировщик состоит из двух частей: энкодера g и декодера f. Энкодер переводит входной сигнал в его представление (код): h=g(x), а декодер восстанавливает сигнал по его коду: x=f(h).

Автокодировщик, изменяя f и g, стремится выучить тождественную функцию x=f(g(x)), минимизируя какой-то функционал ошибки. L(x,f(g(x)))

При этом семейства функций энкодера g и декодера f как-то ограничены, чтобы автоэнкодер был вынужден отбирать наиболее важные свойства сигнала.

Автокодировщик можно использовать для предобучения, например, когда стоит задача классификации, а размеченных пар слишком мало. Или для понижения размерности в данных для последующей визуализации. Либо когда просто надо научиться различать полезные свойства входного сигнала.

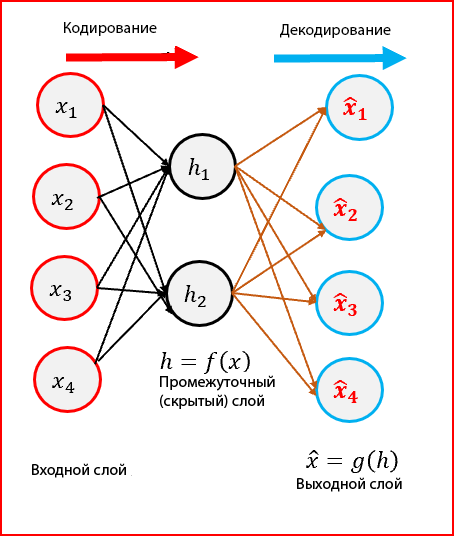


Рисунок 2. Автокодировщик