СДЕЛАНО: Маскированный автоэнкодер для оценки распределения

Матье Жермен MATHIEU.GERMAIN2@USHERBROOKE.CA
Университет Шербрука, Канада

Кароль Грегор КАРОЛ.ГРЕГОР@GMAIL.COM
Google DeepMind

Иэн Мюррей I.MURRAY@ED.AC.UK
Эдинбургский университет, Соединенное Королевство

Хьюго Ларошель ГЮГО.ЛАРОШЕЛЬ@USHERBROOKE.CA
Университет Шербрука, Канада

Абстрактный

В последнее время наблюдается большой интерес к дизайну. модели нейронных сетей для оценки распределения из набора примеров. Мы вводим простой модификация для нейронных сетей автоэнкодера, которая дает мощные генеративные модели. Наш метод маскирует параметры автоэнкодера для соблюдения авторегрессионные ограничения: каждый вход восстанавливается только из предыдущих входов в заданном порядке. Таким образом, автоэнкодер выходы можно интерпретировать как набор условных вероятностей, а их произведение — полная совместная вероятность . Мы также можем обучить одну сеть, которая можно разложить совместную вероятность на несколько разные заказы. Наша простая структура может быть применяется к нескольким архитектурам, включая глубокие те. Векторизованные реализации, такие как на Графические процессоры просты и быстры. Эксперименты показывают, что этот подход конкурентоспособен с современными оценщиками податливого распределения. На тесте время, метод значительно быстрее и масштабируется лучше, чем другие авторегрессионные оценки.

1. Введение

Оценка распределения — это задача оценки совместного распределения p (x) по набору примеров {x (t)} T $_{t=1,}$ который определение общей проблемы. Многие задачи машинного обучения можно сформулировать как изучение только определенных свойств объекта. совместная раздача. Таким образом, хороший оценщик распределения может использоваться во многих сценариях, включая классификацию (Schmah et al., 2009), шумоподавление или отсутствующее вменение входных данных (Пун и Домингос, 2011 г.; Dinh et al., 2014), данные (например, речь) синтез (Uria et al., 2015) и многие другие. Очень характер оценки распределения также делает его особым задача для машинного обучения. По сути, проклятие размерность имеет явное влияние, потому что, поскольку число размеров входного пространства x растет, объем пространство, в котором модель должна дать хороший ответ на p(x) экспоненциально возрастает.

К счастью, недавние исследования добились значительного прогресса. на этой задаче. В частности, были предложены алгоритмы обучения для различных моделей нейронных сетей (Bengio и Бенжио, 2000; Ларошель и Мюррей, 2011 г .; Грегор и Лекун, 2011 г.; Урия и др., 2013; 2014; Кингма и Веллинг, 2014; Резенде и др., 2014; Бенжио и др., 2014 г.; Грегор и др., 2014; Гудфеллоу и др., 2014 г.; Дин и др., 2014). Эти алгоритмы демонстрируют большой потенциал в масштабировании до задачи оценки распределений высокой размерности. В этом В работе мы сосредоточим внимание на авторегрессионных моделях (раздел 3). Точное вычисление p(x) для тестового примера x совместим с этими моделями. Тем не менее, вычислительная стоимость этой операции все еще больше, чем типичные прогнозы нейронной сети для D-мерного входа. Для предыдущих модели глубокой авторегрессии, оценивающие стоимость p(x) O(D) раз больше, чем простой предсказатель точек нейронной сети.

Вклад этой статьи состоит в том, чтобы описать и исследовать простой способ адаптации нейронных сетей автоэнкодера, который делает их конкурентоспособные податливые оценщики распределения, которые быстрее существующих альтернатив. Мы покажем, как замаскировать взвешенные соединения стандартного автоэнкодера для его преобразования в оценщик распределения. Главное использовать маски разработан таким образом, что вывод является авторегрессивным для данный порядок входных данных, т. е. что каждое входное измерение реконструируется исключительно из измерений, предшествующих ему в

Материалы 32 -й Международной конференции по машиностроению Обучение, Лилль, Франция, 2015 г. JMLR: W&CP, том 37.

заказ. Полученный в результате оценщик распределения маскированного автоэнкодера (MADE) сохраняет эффективность одного прохода через обычный автоэнкодер. Реализация на графическом процессоре проста, что делает метод масштабируемым.

Версия MADE с одним скрытым слоем соответствует ранее предложенной авторегрессионной нейронной сети Bengio & Bengio (2000). Здесь мы идем дальше, исследуя глубокие варианты модели. Мы также изучаем обучение MADE для одновременной работы с несколькими порядками входных наблюдений и структур связности скрытых слоев.

Мы тестируем эти расширения на ряде наборов бинарных данных с сотнями измерений и сравниваем их статистическую производительность и масштабирование с сопоставимыми методами.

2. Автоэнкодеры

Краткое описание базового автоэнкодера, на котором основана эта работа, необходимо для ясного понимания того, что следует далее. В этой статье мы предполагаем, что нам дан обучающий набор примеров $\{x^{(t)}\}_{t=1}^{T}$. Мы сосредоточимся на случае бинарных наблюдений, где для каждого D-мерного входа x каждое входное измерение x_d принадлежит {0, 1 }. Мотивация состоит в том, чтобы изучить скрытые представления входных данных, которые раскрывают статистическую структуру распределения, которое их породило.

Автоэнкодер пытается получить прямое скрытое представление h(x) своего входа x таким образом, чтобы из него мы могли получить реконструкцию $\hat{x}$, максимально близкую к x. В частности, у нас есть

$$h(x) = g(b + Wx) \quad (1)$$

$$\hat{x} = \mathrm{sigm}(c + Vh(x)), \quad (2)$$

где W и V — матрицы, b и c — векторы, g — нелинейная функция активации и $\mathrm{sigm}(a) = 1/(1 + \exp(-a))$. Таким образом, W представляет связи от входа к скрытому слою, а V представляет связи от скрытого к выходному слою.

Чтобы обучить автоэнкодер, мы должны сначала указать функцию потерь при обучении. Для бинарных наблюдений естественным выбором является потеря кросс-энтропии:

$$\ell(x) = \sum_{d=1}^{D} -x_d \log \hat{x}_d - (1 - x_d) \log(1 - \hat{x}_d) . \quad (3)$$

Рассматривая $\hat{x}_d$ как модельную вероятность того, что x_d равно 1, перекрестную энтропию можно понимать как принимающую форму отрицательной логарифмической функции правдоподобия. Обучение автоэнкодера соответствует оптимизации параметров {W, V, b, c} для уменьшения средних потерь на обучающих примерах, обычно с (мини-пакетным) стохастическим градиентным спуском.

Одним из преимуществ парадигмы автоэнкодера является ее гибкость. В частности, получить кодер с глубоким автоматическим кодированием несложно, вставив дополнительные скрытые слои между входными данными. и выходные слои. Его главный недостаток заключается в том, что представление, которое он изучает, может быть тривиальным. Например, если скрытый слой не меньше входного, каждый из скрытых элементов может научиться «копировать» одно входное измерение, чтобы точно реконструировать все входные данные на выходном слое. Одним из очевидных следствий этого наблюдения является то, что функция потерь в уравнении 3 на самом деле не является правильной функцией логарифмического правдоподобия . В самом деле, поскольку идеальная реконструкция могла быть достигнута, $\prod_d \hat{x}_d^{x_d}(1-\hat{x}_d)^{1-x_d}$ предполагаемое «распределение» данных q(x)=Q можно было бы узнать равным 1 для любого x и, таким образом, не нормализовано должным образом ($\sum_x q(x) \neq 1$).

3. Оценка распределения как авторегрессия

Интересный вопрос заключается в том, какое свойство мы могли бы наложить на автоэнкодер, чтобы его выходные данные можно было использовать для получения достоверных вероятностей. В частности, мы хотели бы иметь возможность записывать p(x) таким образом, чтобы его можно было вычислить на основе вывода правильно исправленного автоэнкодера.

Во-первых, мы можем использовать тот факт, что для любого распределения правило произведения вероятностей подразумевает, что мы всегда можем разложить его на произведение его вложенных условных выражений.

$$p(x) = \prod_{d=1}^{D} p(x_d \mid x_{<d}), \quad (4)$$

где $x_{<d} = [x_1, \ldots , x_{d-1}]^\top$.

Определяя $p(x_d = 1 \mid x_{<d}) = \hat{x}_d$, и, таким образом , $p(x_d = 0 \mid x_{<d}) = 1 - \hat{x}_d$, потеря уравнения 3 становится действительной отрицательной логарифмической вероятностью:

$$\begin{aligned} -\log p(x) &= \sum_{d=1}^{D} -\log p(x_d \mid x_{<d}) \\ &= \sum_{d=1}^{D} -x_d \log p(x_d = 1 \mid x_{<d}) \\ &\quad - (1 - x_d) \log p(x_d = 0 \mid x_{<d}) = \ell(x). \end{aligned} \quad (5)$$

Это подключение позволяет определить автокодировщики, которые можно использовать для оценки распределения. Каждый выход $\hat{x}_d = p(x_d \mid x_{<d})$ должен быть функцией, принимающей в качестве входных данных только $x_{<d}$ и выводящей вероятность наблюдения значения x_d th в d - измерении. В частности, автоэнкодер формирует правильное распределение, если каждая выходная единица $\hat{x}_d$ зависит только от предыдущих входных единиц $x_{<d}$, а не от других единиц $x_{\geq d} = [x_d, \ldots , x_D]^\top$.

Мы называем это свойство свойством авторегрессии, поскольку вычисление отрицательного логарифмического правдоподобия (5) эквивалентно последовательному прогнозированию (регрессии) каждого измерения входных данных x.

4. Маскированные автоэнкодеры

Теперь вопрос заключается в том, как модифицировать автоэнкодер, чтобы он удовлетворял свойству авторегрессии. Поскольку выход $\hat{x}_d$ должен зависеть только от предшествующих входов $x_{<d}$, это означает, что не должно быть вычислительного пути между выходным блоком $\hat{x}_d$ и любым из входных блоков $x_d, \ldots, x_D$. Другими словами, для каждого соединения хотя бы одно соединение (в матрице W или V) должно быть равно 0.

Удобный способ обнуления соединений состоит в том, чтобы поэлементно умножить каждую матрицу на двоичную матрицу масок, элементы которой равны 0 и соответствуют соединениям, которые мы хотим удалить. Для автоэнкодера с одним скрытым слоем мы пишем

$$h(x) = g(b + (W \odot MW)x) \quad (6)$$

$$\hat{x} = \mathrm{sigm}(c + (V \odot MV)h(x)) \quad (7)$$

где MW и MV — маски для W и V соответственно. Таким образом, маски MW и MV удовлетворяют свойству авторегрессии.

Чтобы наложить свойство авторегрессии, мы сначала присваиваем каждой единице в скрытом слое целое число m от 1 до D-1 — номера максимальное количество входных единиц, к которым скрытый элемент может быть подключен. Мы запрещаем $m(k)=D$, так как эта скрытая единица будет зависеть от всех входных данных и не может быть использована при моделировании любого из условных выражений $p(x_d \mid x_{<d})$. Точно так же мы исключаем $m(k)=0$, так как это создало бы постоянные скрытые единицы.

Ограничения на максимальное количество входов в каждую скрытую единицу закодированы в матрице, маскирующей связи между входной и скрытой единицами:

$$M^{V}_{k,d} = 1_{m(k) \geq d} = \begin{cases} 1, & \text{если } m(k) \geq d \\ 0 & \text{в противном случае,} \end{cases} \quad (8)$$

для $d \in \{1, \ldots, D\}$ и $k \in \{1, \ldots, K\}$. В целом, нам нужно th для кодирования ограничения, согласно которому выходной блок d связан только с $x_{<d}$ (и, следовательно, не с x_d). Следовательно, выходные веса могут связывать выход d только со скрытыми единицами с $m(k) < d$, т. е. единицами, которые связаны не более чем с $d-1$ входными единицами. Эти ограничения закодированы в матрице выходной маски:

$$M^{W}_{d,k} = 1_{d > m(k)} = \begin{cases} 1, & \text{если } d > m(k) \\ 0 & \text{в противном случае,} \end{cases} \quad (9)$$

снова для $d \in \{1, \ldots, D\}$ и $k \in \{1, \ldots, K\}$. Обратите внимание, что согласно этому правилу никакие скрытые единицы не будут подключены к первой выходной единице $\hat{x}_1$, как это и требовалось.

Из этих конструкций масок мы можем легко продемонстрировать, что соответствующий маскированный автоэнкодер удовлетворяет свойству авторегрессии. Во-первых, отметим, что, поскольку маски MV и MW представляют связность сети, их матричное произведение $M^{V,W} = MV MW$ представляет связность между входным и выходным слоями. Конкретно, $M^{V,W}_{d0,d}$ — количество сетевых путей между выходной единицей $\hat{x}_{d0}$ и входной единицей x_d. Таким образом, чтобы продемонстрировать свойство авторегрессии, нам нужно показать, что $M^{V,W}$ является строго нижней диагональю, т. е. $M^{V,W}_{d0,d} = 0$, если $d0 \geq d$. По определению

$$M^{V,W}_{d0,d} = \sum_{k=1}^{K} M^{V}_{d0,k} M^{W}_{k,d} = \sum_{k=1}^{K} 1_{d0 > m(k)} 1_{m(k) \geq d} \quad (10)$$

Если $d0 \leq d$, то не существует таких значений $m(k)$, которые одновременно были бы строго меньше d и больше или равны $d0$. Таким образом, $M^{V,W}_{d0,d} = 0$.

Построение масок MV и MW требует только присвоения значений $m(k)$ каждому скрытому элементу. Можно представить себе попытку присвоить (приблизительно) равное количество единиц каждому допустимому значению $m(k)$. Вместо этого в наших экспериментах мы устанавливаем $m(k)$ путем выборки из равномерного дискретного распределения, заданного целыми числами от 1 до $D-1$, независимо для каждой из K скрытых единиц.

Предыдущие работы по авторегрессионным нейронным сетям также показали преимущество использования прямых связей между входным и выходным слоями (Bengio & Bengio, 2000). В этом контексте реконструкция становится:

$$\hat{x} = \mathrm{sigm}(c + (V \odot MV)h(x) + (A \odot MA)x), \quad (11)$$

где A — матрица связи параметра, а MA — его матрица маски. Чтобы удовлетворить свойству авторегрессии, MA просто должна быть строго нижней диагональной матрицей, в противном случае заполненной единицами. Такие прямые связи мы использовали и в наших экспериментах.

4.1. Глубоко СДЕЛАНО

Одним из преимуществ фреймворка маскированного автоэнкодера, описанного в предыдущем разделе, является то, что он естественным образом обобщается на глубокие архитектуры. Действительно, как мы увидим, назначая максимальное количество подключенных входов всем элементам глубокой сети, маски могут быть построены аналогичным образом, чтобы удовлетворять свойству авторегрессии.

Для сетей со скрытыми слоями $L>1$ мы используем верхние индексы для индексации слоев. Матрица первого скрытого слоя (ранее W) будет обозначаться как W^1, матрица второго как W^2 и так далее. Количество скрытых единиц (ранее K) в каждом скрытом слое будет аналогично индексироваться как K^l, где l — индекс скрытого слоя. Мы также обобщим обозначение максимального числа подключенных входов th th блока k в l слой мл (к).

Мы уже обсудили, как определить матрицу маски первого слоя th таким образом, чтобы она гарантировала, что ее элемент k подключен к не более m(k) (теперь $m^1(k)$) входов. Чтобы применить то же свойство ко второму скрытому слою, мы должны просто убедиться , что каждая единица k связана только с единицами первого слоя.

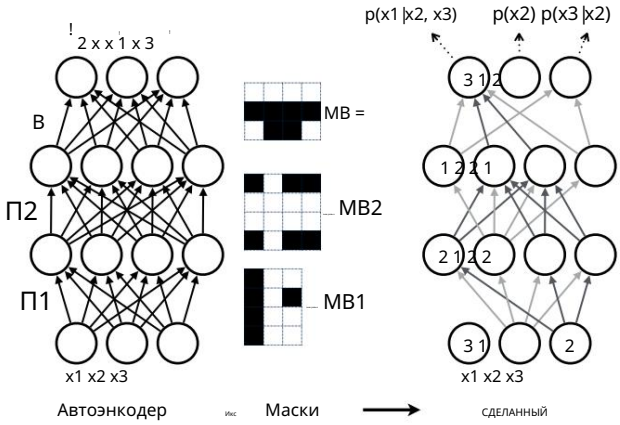

Рис. 1. Слева: обычный автоэнкодер с тремя скрытыми слоями. Ввод внизу проходит через полносвязные слои и точечные нелинейности. В последнем верхнем слое производится реконструкция, определяемая как распределение вероятностей по входным данным. Поскольку это распределение зависит от самих входных данных, стандартный автоматический кодировщик не может предсказывать или выбирать новые данные. Правильно: СДЕЛАНО. Сеть имеет ту же структуру, что и автоэнкодер, но набор связей удаляется таким образом, что каждая входная единица только предсказывается из предыдущих с использованием мультипликативных масок. Цифры в скрытых единицах обозначают максимальное количество входных данных, на основе которых построено k на основе этих чисел (см. уравнения 12 и 13). Эти маски гарантируют, что MADE удовлетворяет свойству авторегрессии , позволяя формировать вероятностную модель, в этом примере p(x) = p(x2) p(x3 |x2) p(x1 |x2, x3). Соединения светло-серого цвета соответствуют путям, которые зависят только от 1 входа, а соединения темно- серого цвета зависят от 2 входов.

связаны не более чем с m2 (k, ^{0}) входы, т.е. единицы первого слоя для которых m1 (k) m2 (k ^{0}).

Это правило можно обобщить на любой слой l следующим образом:

$$MB^{l}_{k0,k} = 1ml(k0) \quad ml \quad 1(k) = \begin{cases} 1, \text{ если мл (к } ^{0}) \quad мл \quad 1 \text{ (к)} \\ 0 \text{ иначе.} \end{cases} \tag{12}$$

Кроме того, принимая l = 0 за входной слой и определяя единицу m0 (d) = d (что интуитивно понятно, так как d ^{й} ввода в действие берет свои значения из d первых входных данных), это определение также применимо для весов первого скрытого слоя. Что касается выходной маски, нам просто нужно адаптировать ее определение, используя ограничения связности последнего скрытого слоя mL(k) вместо первого:

$$MB^{д}_{,к} = 1d>mL(k) = \begin{cases} 1, \text{ если } d > mL(k) \\ 0 \text{ иначе.} \end{cases} \tag{13}$$

Как и в случае с одним скрытым слоем, значения ml (k) для каждого скрытого слоя l {1, . . . , L} выбираются равномерно. Чтобы избежать неподключенных единиц, значение для мл (k) выбирается быть больше или равно минимальной связности на предыдущем слое, т. е. mink0 ml 1 (k ^{0}).

4.2. Независимое от порядка обучение

До сих пор мы предполагали, что условные операторы, моделируемые MADE, согласуются с естественным порядком измерений x. Однако нас может заинтересовать моделирование условий, связанных с произвольным порядком измерений входных данных.

В частности, Uria et al. (2014) показали, что обучение авторегрессионной модели для всех порядков может быть полезным. Мы называем этот подход обучением, не зависящим от порядка. Этого можно достичь путем выборки порядка перед каждым стохастическим/минипакетным градиентным обновлением модели. У этого подхода есть два преимущества. Во- первых, отсутствующие значения в частично наблюдаемых входных векторах могут быть эффективно введены: мы вызываем упорядочение, при котором все наблюдаемые измерения предшествуют ненаблюдаемым, что упрощает вывод. Во-вторых, ансамбль авторегрессионных моделей может быть построен «на лету», используя тот факт, что условные операторы для двух разных порядков не гарантируют точного соответствия (и, таким образом, технически соответствуют слегка отличающимся моделям). Затем ансамбль легко получить путем выборки набора порядков, вычисления вероятности x при каждом упорядочении и усреднении.

Удобно, что в MADE порядок просто представлен вектором m0 = [m0 (1), . . . , м0 (Д)]. В частности, m0 (d) соответствует положению ^{й} размер произведении кондиционалов. Таким образом, случайное упорядочение может быть получено путем случайной перестановки упорядоченного вектора [1, . . . , Д]. Затем из этих значений каждого m0 можно создать матрицу маски первого, В скрытого слоя. Во время независимого от порядка обучения повторной случайной перестановки последнего значения m0 достаточно, чтобы получить новый случайный порядок.

4.3. Обучение, не зависящее от подключения

Одним из преимуществ независимого от порядка обучения является то, что оно эффективно позволяет нам обучать столько моделей, сколько существует порядков, используя общий набор параметров. Это можно использовать , создавая ансамбли моделей во время тестирования.

В MADE, в дополнение к выбору порядка, мы также должны выбрать ограничение связности каждой скрытой единицы ml (k). Таким образом, мы могли бы обучать визуализации MADE, чтобы он также не зависел от шаблона подключения, созданного этими ограничениями. Чтобы добиться этого, вместо выборки значений ml (k) для всех блоков и слоев раз и навсегда перед обучением, мы фактически перевыбираем их для каждого обучающего примера или мини-пакета. Это по-прежнему практично, так как операцию создания масок легко распараллелить. Обозначая [ml(1), ml.= , ml (Kl)], и предполагая поэлементную и параллельную реализацию операции 1a b для векторов, таких, что 1a b есть матрица

Алгоритм 1 Вычисление p(x) и градиенты обучения для MADE с выборкой по порядку и связности. D — размер входных данных, L — количество скрытых слоев и K — количество скрытых единиц.

Вход: обучающий вектор наблюдения x
Выход: p(x) и градиенты log p(x) по параметрам .

Выборка векторов ml m0
перемешать([1, . . . , D]) для
l от 1 до L do
для k от 1 до Kl do ml (k)
Uniform([mink0 ml 1 (k ⁰), . . . , D 1])
конец за
конец за

Построение масок для каждого слоя
для l от 1 до L do
MWl 1мл мл-1
конец для

MB 1m0>мл

Вычисление p(x) 0
ч (x) x
для l от 1 до L do
л ч (x) g(b end л + (Вл МВл)ч л 1 (Икс))
for xb sigm(c +
(V MV)h L(x)) p(x) exp PD d=1 xd log xbd
+ (1 xd) log(1 xbd)

Вычисление градиентов log p(x) tmp
xb x δc tmp δV tmp h L(x)
> MB tmp
(tmp>(V MB))>
для л от л до 1 сделать
tmp tmp g δb ⁰(б л + (Вл МВл)ч л 1 (Икс))
tmp δWl tmp h
tmp (tmp>(Wl
MWl))> л 1 > МВл (x)

конец
для возврата p(x), δb 1 . . . , δb L, δW1 , . . . , δWL, δc, δV

чей элемент i, j равен 1ai bj , то маски скрытого слоя просто MWl = 1ml ml 1 .

Благодаря повторной выборке подключений скрытых единиц для каждого обновления, каждая скрытая единица будет иметь постоянно изменяющееся количество входящих входных данных во время обучения. Однако отсутствие связи неотличимо от инстанцированной связи с единицей с нулевым значением, что может сбить нейронную сеть с толку во время обучения. В аналогичной ситуации Uria et al. (2014) сообщил каждой скрытой единице, какой процент

вводили бинарные индикаторные переменные, связанные с дополнительными обучаемыми весами. Мы рассмотрели возможность применения аналогичной стратегии, используя MWl , но оказалось, что матрица весов компаньона , которая также маскируется постоянный однозначный вектор:

час л (x) = г(б л + (Вл МВл)ч л 1 (x) + (UlMWl)1) (14)

Применялась и аналогичная параметризация выходного слоя . Эти веса, определяющие связность, были полезны лишь иногда. В наших экспериментах мы относились к выбору их использования как к гиперпараметру.

Более того, в ходе наших экспериментов мы обнаружили, что маски выборки для каждого примера иногда могут привести к чрезмерной регуляризации MADE и спровоцировать недообучение. Чтобы решить эту проблему, мы также рассмотрели выборку только из конечного списка масок. Во время обучения MADE циклически просматривает этот список, используя по одному для каждого обновления. Затем во время тестирования мы усредняем вероятности, полученные для всех масок в списке.

Алгоритм 1 подробно описывает, как p(x) вычисляется MADE, а также как получить градиент `(x) для обучения стохастическому градиентному спуску. Для простоты псевдокод предполагает обучение, не зависящее от порядка и связности, не предполагает использования условных весовых матриц или прямых соединений ввода/вывода. Рисунок 1 также иллюстрирует пример такой двухслойной сети MADE вместе с ее значениями ml (k) и ее масками.

5. Связанная работа

В последнее время было проведено много работ по изучению использования нейронных сетей с прямой связью, подобных автоэнкодеру, в качестве вероятностных генеративных моделей. Частично мотивация этого исследования заключается в проверке распространенного предположения о том, что использование моделей с вероятностными скрытыми переменными и управляемыми статистическими суммами (такими как ограниченная машина Больцмана (Салахутдинов и Мюррей, 2008)) является неизбежным злом в разработка мощных генеративных моделей для многомерных данных.

Работа над нейронной авторегрессионной оценкой распределения или NADE (Larochelle & Murray, 2011) продемонстрировала, что архитектуры с прямой связью могут фактически использоваться для формирования современных и даже управляемых оценок распределения.

Недавно было предложено глубокое расширение NADE, еще больше улучшившее современное состояние оценки распределения (Uria et al., 2014). В этой работе была представлена рандомизированная процедура обучения, которая (как и MADE) имеет почти такую же стоимость за итерацию, что и стандартный автоэнкодер. К сожалению, глубокие модели NADE по-прежнему требуют D проходов прямой связи через сеть для оценки вероятности p(x) D -мерного тестового вектора. Вычисление как каждый скрытого слоя может быть распределено между этими проходами.

Таблица 1. Сложность различных моделей в таблице 6 для расчета точный тест отрицательной логарифмической вероятности. R - количество заказов используется, D — входной размер, а K — размер скрытого слоя (при условии, что скрытые слои одинакового размера).

Модель	ОНЛЛ
RBM 25 CD шагов	$O(\min(2DK, D2K))$
DARN	$O(2KD)$
NADE (фиксированный порядок)	$O(HD)$
EoNADE 1гл, р орд.	$O(RDK)$
EoNADE 2гл, р орд.	$O(RDK^2)$
СДЕЛАНО 1гл, 1 порция.	$O(HD+D^2)$
СДЕЛАНО 2гл, 1 изд.	$O(DK+K2+D2)$
СДЕЛАНО 1гл, р орд.	$O(R(HD+D^2))$
СДЕЛАНО 2гл, р орд.	$O(R(DK+K2+D2))$

Таблица 2. Количество входных измерений и количество примеров в поезд, валидация и тестовые сплиты.

Имя	# Входы	Поезд действителен.	Тест
Взрослый	123	5000	1414 26147
Подключить4	126	16000	4000 47557
ДНК	180	1400	600 1186
Грибы	112	2000	500 5624
НИПС-0-12	500	400	100 1240
OCR-буквы	128	32152	10000 10000
RCV1	150	40000	10000 150000
Интернет	300	14000	3188 32561

хотя на практике это медленнее, чем оценка одного прохода в стандартном автоэнкодере. В глубоких сетях со скрытым K единиц на слой, оценка тестового вектора стоит $O(DK^2)$.

Глубокие авторегрессивные сети (DARN, Грегор и др., 2014), также предоставляют вероятностные модели с примерно те же затраты на обучение, что и у стандартных автоэнкодеров. Латентное представление DARN состоит из двоичных, стохастических скрытых единиц. Хотя моделирование с помощью этих моделей выполняется быстро, оценка точные тестовые вероятности требуют суммирования по всем конфигурациям скрытого представления, которое является экспоненциальным в вычислении. Таким образом, рекомендуется приближение Монте-Карло .

Основное преимущество MADE состоит в том, что оценка вероятностей сохраняет эффективность автоэнкодеров с небольшими потерями. дополнительные затраты на простые операции маскирования. В таблице 1 перечислены вычислительная сложность точного расчета вероятностей для различных моделей. DARN и RBM экспоненциальны по размерности скрытых данных или данных, тогда как НАДЕ и СДЕЛАНО полиномиальны. СДЕЛАНО требует только один проход через автоэнкодер, а не D - проходы требуется НАДЭ. На практике мы также наблюдаем, что однослойный MADE на порядок быстрее, чем однослойный NADE, для того же размера скрытого слоя, несмотря на Совместное вычисление NADE для получения той же асимптотики масштабирование. Вычисления NADE не могут быть векторизованы как эффективно. У глубоких версий MADE также лучше масштабирование, чем NADE во время тестирования. Стоимость обучения для MADE, DARN и deep NADE будут похожи.

Перед работой над NADE Bengio & Bengio (2000) предложили архитектуру нейронной сети, соответствующую частный случай модели MADE с одним скрытым слоем, без рандомизация порядка ввода и подключения. Вклад нашей работы заключается в том, чтобы выйти за рамки этого особого случая, исследуя глубокие варианты и обучение, не зависящее от порядка/связности

Интересная интерпретация авторегрессионной выборки по маске — это структурированная форма регуляризации отсева (Srivasava et al., 2014). В частности, он имеет сходство с маскировка в сетях dropconnect (Wan et al., 2013). Исключением является то, что сгенерированные здесь маски должны гарантировать свойство авторегрессии автоэнкодера, в то время как в Ван и др. (2013), каждый элемент в маске генерируется независимо.

6. Эксперименты

Чтобы проверить производительность нашей модели, мы рассмотрели два разных теста: набор двоичных файлов UCI наборы данных и бинаризованный набор данных MNIST. Код для Воспроизвести эксперименты из этой статьи можно по адресу https://github.com/mgermain/MADE/releases/tag/ICML2015. Представленные здесь результаты представляют собой среднюю отрицательную логарифмическую вероятность в тестовом наборе каждого соответствующего набора данных. Все эксперименты проводились с использованием стохастического градиентного спуска (SGD) с мини-пакетами размером 100 и опережением 30 за раннюю остановку.

6.1. Комплект для оценки UCI

Мы используем двоичный оценочный пакет UCI, который был впервые поставлен вместе в Ларошель и Мюррей (2011). это коллекция из 7 относительно небольших наборов данных из Калифорнийского университета, репозитория машинного обучения Ирвина и писем OCR набор данных Стэнфордской лаборатории искусственного интеллекта. Таблица 2 дает обзор масштаба этих наборов данных и способа их разделения.

Эксперименты проводились с сетями по 500 единиц в секунду. скрытый слой, используя обновление обучения adadelta (Zeiler, 2012) с затуханием 0,95. Другие гиперпараметры варьировались, как показано в таблице 3 . Отмечаем в качестве количества масок количество различных масок, через которые СДЕЛАНО циклы во время обучения. В безграничном случае маски выбираются на лету и никогда явно не используется повторно, если только шанс. В этой ситуации во время проверки и тестирования 300 и 1000 выборочных масок используются для усреднения вероятностей.

Таблица 3. Поиск по сетке UCI

Гиперпараметр	Пробные значения
# Скрытый слой	1, 2
Функция активации	ReLU, Softplus
Ададельта эпсилон	10-5 10-7 10-9
Условные веса Правда, Ложь Количество заказов	1, 8, 16, 32, без ограничений

Таблица 5. Бинаризованный поиск по сетке MNIST

Пробные значения гиперпараметров	
# Скрытый слой 1, 2	
Скорость обучения	0,1, 0,05, 0,01, 0,005 1, 2, 4,
# масок	8, 16, 32, 64

Результаты представлены в таблице 4. Мы видим, что MADE является одной из лучших моделей для половины наборов данных и конкурентоспособна в остальном. Чтобы уменьшить беспорядок, мы не сообщали о стандартных отклонениях, которые были довольно небольшими и одинаковыми для всех моделей. Однако для полноты мы приводим стандартные отклонения в отдельной таблице в дополнительных материалах.

Анализ гиперпараметров, выбранных для каждого набора данных , не выявил явного победителя. Однако из Таблицы 4 мы видим , что когда выборка по маске помогает, она помогает совсем немного, а когда нет, влияние незначительно на все буквы, кроме OCR. Еще одно интересное замечание заключается в том, что кондиционирующие веса почти не повлияли, за исключением NIPS-0-12, где они помогли.

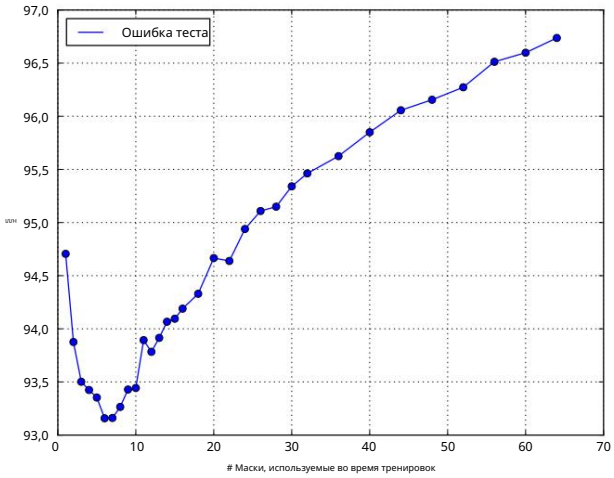

Рисунок 2. Влияние количества масок, используемых с одним скрытым слоем, сетью из 500 скрытых единиц, на бинаризованный MNIST.

6.2. Бинаризованная оценка MNIST

Мы использовали версию MNIST, бинаризованную Салахутдиновым и Мюрреем (2008). MNIST — это набор из 70 000 рукописных цифр размером 28×28 пикселей. Мы используем то же разделение , что и в Larochelle & Murray (2011), состоящее из 50 000 для обучающего набора, 10 000 для проверочного набора и 10 000 для тестового набора.

Эксперименты проводились с использованием обновления обучения adagrad (Duchi et al., 2010) с эпсилон 10 6 . Поскольку MADE намного эффективнее, чем NADE, мы решили изменить размер скрытого слоя от 500 до 8000 единиц. Видя , что увеличение числа единиц всегда помогает, мы использовали 8000. Даже с таким большим скрытым слоем наша реализация MADE на GPU была достаточно эффективной. При использовании одной маски одна эпоха обучения требует около 14 и 44 секунд для одного скрытого слоя и двух скрытых слоев СДЕЛАНО соответственно. При использовании 32 выборочных масок время обучения увеличивается до 33 и 100 соответственно. Все эти тайминги меньше, чем наша реализация GPU модели NADE с 500 скрытыми единицами, которая требует около 130 секунд на эпоху.

Эти тайминги были получены на графическом процессоре NVIDIA K20.

Основываясь на том, что мы узнали из экспериментов UCI, мы установили функцию активации ReLU, а кондиционирующие веса не использовались. Варьируемые гиперпараметры представлены в таблице 5.

Результаты представлены в таблице 6 вместе с другими результатами . взято из литературы. Опять же, несмотря на свою управляемость, MADE конкурентоспособен с другими моделями. Следует отметить тот факт, что лучшая модель MADE превосходит однослойную сеть NADE, которая в остальном была лучшей моделью среди моделей , требующих только одного прохода с прямой связью для вычисления вероятностей журнала.

В этих экспериментах мы ясно наблюдали явление чрезмерной регуляризации из-за использования слишком большого количества масок. Когда использовалось более четырех порядков, более глубокий вариант MADE всегда давал лучшие результаты. Для двухслойной модели добавление масок во время обучения помогло увеличить число до 64, после чего отрицательная логарифмическая вероятность начала увеличиваться. Мы наблюдали аналогичную картину для однослойной модели, но в этом случае падение составило около 8 масок. Рисунок 2 более точно иллюстрирует это поведение для одного слоя MADE с 500 скрытыми единицами, обученного только изменением количества используемых масок и размера мини-пакетов (83, 100, 128).

Мы случайным образом выбрали 100 цифр из нашей наиболее эффективной модели из таблицы 6 и сравнили их с их ближайшими соседями в обучающем наборе (рис . 3), чтобы убедиться, что сгенерированные выборки не являются простым запоминанием. Каждая строка цифр использует другую маску, которая была видна сетью во время обучения.

Таблица 4. Отрицательные результаты теста логарифмического правдоподобия различных моделей на нескольких наборах данных. Лучший результат, а также любой другой результат с перекрывающийся доверительный интервал показан жирным шрифтом. Обратите внимание, что, поскольку дисперсия DARN не была доступна, мы считали ее равной нулю.

Модель	Adult	Connect4	DNA	Mushrooms	NIPS-0-12	OCR-буквы	RCV1	Web
MoBernoullis 20.44	RBM 16.26	FVSBN	23,41	98,19	14,46	290,02	40,56	47,59	30,16
13.17 NADE (фиксированный заказ)	13.19		22,66	96,74	15,15	277,37	43,05	48,88	29,38
EoNADE 1hl (16 порций)	13.19	DARN	12,39	83,64	10,27	276,88	39,30	49,84	29,35
			11,99	84,81	9,81	273,08	27,22	46,66	28,39
			12,58	82,31	9,69	272,39	27,32	46,12	27,87
			11,91	81,04	9,55	274,68	28,17	46,10	28,83
MADE 13.12	MADE выборка маски		11,90	83,63	9,68	280,25	28.34	47,10	28.53
13.13			11,90	79,66	9,69	277,28	30.04	46,74	28.25

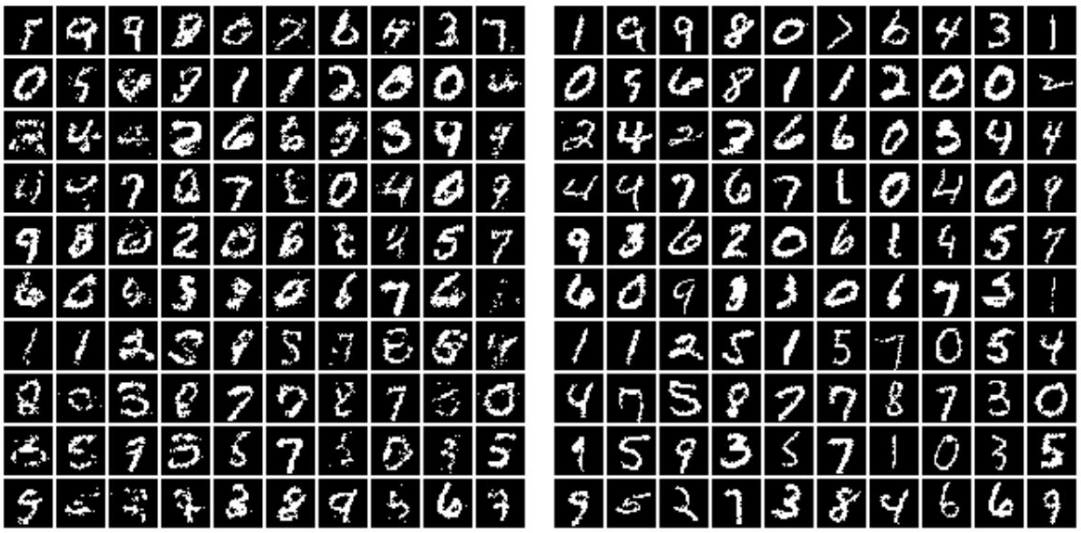

Рисунок 3. Слева: Образцы из 2-х скрытых слоев СДЕЛАНЫ. Справа: ближайший сосед в бинаризованном MNIST.

Таблица 6. Отрицательные результаты теста логарифмического правдоподобия различных моделей на бинарный набор данных MNIST.

Модель	лог р
RBM (500 ч, 25 шагов CD)	86,34
ДБМ 2гл	84,62
ДБН 2гл	84,55
ДАРН nh=500	84,71
DARN nh=500, adaNoise	84,13
МоБернуллис K=10	168,95
МоБернуллис K=500	137,64
НАДЕ 1гл (фиксированный заказ)	88,33
EoNADE 1гл (128 заказов)	87,71
EoNADE 2гл (128 заказов)	85,10
СДЕЛАНО 1гл (1 маска)	88,40
СДЕЛАНО 2гл (1 маска)	89,59
СДЕЛАНО 1гл (32 маски)	88.04
СДЕЛАНО 2гл (32 маски)	86,64

7. Заключение

Мы предложили MADE, простую модификацию автоэнкодеров
что позволяет использовать их в качестве оценок распределения. СДЕЛАННЫЙ
демонстрирует, что можно получить прямые, дешевые оценки
многомерных совместных вероятностей за один проход
через автоэнкодер. Как и стандартные автоэнкодеры, наше расширение легко
векторизовать и реализовать на графических процессорах. СДЕЛАННЫЙ
может оценивать многомерные вероятностные распределения с помощью
лучшее масштабирование, чем раньше, при сохранении современного состояния
статистическая производительность.

Благодарности

Мы благодарим Марка-Алексаńдра Котэ за помощь в реализации NADE в Theano и во всем Theano (Bastien et al., 2012 г.; Bergstra et al., 2010) группа авторов. Мы также
спасибо NSERC, Calcul Quebec и Compute Canada.

использованная литература

Бастьен, Фредерик , Ламблин, Паскаль, Паскану, Разван, Бергстра, Джеймс, Гудфеллоу, Ян Дж., Бержерон, Арно, Бушар, Николя и Бенжио, Йошуа. Theano: новые функции и улучшения скорости. Семинар NIPS 2012 по глубокому обучению и неконтролируемому изучению функций, 2012 г.

Бенжио, Йошуа и Бенжио, Сэми. Моделирование многомерных дискретных данных с помощью многослойных нейронных сетей. В Достижениях в области систем обработки нейронной информации 12 (NIPS 1999), стр. 400–406. Массачусетский технологический институт, 2000.

Бенжио, Йошуа, Лауфер, Эрик, Ален, Гийом и Йосин Ски , Джейсон. Глубокие генеративные стохастические сети, обучаемые обратным распространением. В материалах 31-й ежегодной международной конференции по машинному обучению (ICML 2014), стр. 226–234. JMLR.org, 2014.

Бергстра, Джеймс, Брело, Оливье, Бастьен, Фредерик , Ламблин , Паскаль, Паскану, Разван, Дежарден, Гийом, Туриан, Жозеф, Варде-Фарли, Давид и Бенжио, Йошуа. Theano: компилятор математических выражений CPU и GPU. В материалах конференции Python для научных вычислений (SciPy), июнь 2010 г. Устная презентация.

Дин, Лоран, Крюгер, Дэвид и Бенжио, Йошуа. NICE: нелинейная оценка независимых компонентов, 2014. arXiv:1410.8516v2.

Дучи, Джон, Хазан, Элад и Сингер, Йорам. Адаптивные субградиентные методы онлайн-обучения и стохастической оптимизации. Технический отчет, Департамент EECS, Калифорнийский университет, Беркли, март 2010 г.

Гудфеллоу, Ян, Пуже-Абади, Джин, Мирза, Мехди, Сюй, Бинг, Уорд-Фарли, Дэвид, Озаир, Шерджил, Курвиль, Аарон и Бенжио, Йошуа. Генеративные состязательные сети. В Достижениях в области систем обработки нейронной информации 27 (NIPS 2014), стр. 2672–2680. Карран Ассошиэйтс, Инк., 2014.

Грегор, Кароль и ЛеКун, Янн. Изучение представлений путем максимального сжатия, 2011 г. arXiv: 1108.1169v1.

Грегор, Кароль, Данихелка, Иво, Мних, Андрей, Бланделл, Чарльз и Вирстра, Даан. Глубокая авторегрессивная сеть работает. В материалах 31-й ежегодной международной конференции по машинному обучению (ICML 2014), стр. 1242–1250. JMLR.org, 2014.

Кингма, Дидерик П. и Веллинг, Макс. Автокодирование вариационного байеса. В материалах 2-й Международной конференции по представительствам в обучении (ICLR 2014), 2014 г.

Ларошель, Хьюго и Мюррей, Иэн. Нейронная авторегрессионная оценка распределения. В материалах 14-й Международной конференции по искусственному интеллекту и статистике (AISTATS 2011), том 15, стр. 29–37, Ft. Лодердейл, США, 2011 г. JMLR W&CP.

Пун, Хойфунг и Домингос, Педро. Сеть сумм-продуктов: новая глубокая архитектура. В материалах 20-й конференции по неопределенности в искусственном интеллекте (UAI 2011), стр. 337–346, 2011 г.

Резенде, Данило Хименес, Мохамед, Шакир и Вирстра, Даан. Стохастическое обратное распространение и приближенный вывод в глубоких генеративных моделях. В материалах 31-й Международной конференции по машинному обучению (ICML 2014), стр. 1278–1286, 2014 г.

Салахутдинов, Руслан и Мюррей, Иэн. О количественном анализе сетей глубокого доверия. В материалах 25-й ежегодной международной конференции по машинному обучению (ICML 2008), стр. 872–879. Омнипресс, 2008.

Шмах, Таня, Хинтон, Джеффри Э., Земель, Ричард С., Смолл, Стивен Л. и Стротер, Стивен С. Генеративное и дискриминативное обучение RBM для классификации изображений фМРТ. В Достижениях в системах обработки нейронной информации 21 (NIPS 2008), стр. 1409–1416, 2009.

Сривастава, Нитиш, Хинтон, Джеффри, Крижевский, Алекс, Суцкевер, Илья, и Салахутдинов, Руслан. Dropout: простой способ предотвратить переоснащение нейронных сетей. Журнал исследований машинного обучения, 15:1929–1958, 2014 г.

Урия, Бениньо, Мюррей, Иэн, и Ларошель, Хьюго. RNADE: нейронная авторегрессионная оценка плотности с действительным знаком. В Достижениях в области систем обработки нейронной информации 26 (NIPS 2013), стр. 2175–2183, 2013.

Урия, Бениньо, Мюррей, Иэн, и Ларошель, Хьюго. Глубокий и удобный оценщик плотности. В материалах 31-й Международной конференции по машинному обучению (ICML 2014), стр. 467–475, 2014 г.

Урия, Бениньо, Мюррей, Иэн, Реналс, Стив и Валентини Ботиньян, Кассия. Моделирование акустических зависимостей с помощью искусственных нейронных сетей: Trajectory-RNADE. В материалах 40-й Международной конференции IEEE по акустике, обработке речи и сигналов (ICASSP 2015). Общество обработки сигналов IEEE, 2015 г.

Ван, Ли, Зейлер, Мэтью Д., Чжан, Сиксин, Лекун, Янн и Фергус, Роб. Регуляризация нейронных сетей с помощью dropconnect. В материалах 30-й Международной конференции по машинному обучению (ICML 2013), стр. 1058–1066, 2013 г.

Зейлер, Мэтью Д. АДАДЕЛЬТА: метод адаптивной скорости обучения, 2012. arXiv: 1212.5701v1.

Machine Translated by Google

СДЕЛАНО: Маскированный автоэнкодер для оценки распределения. Дополнительный материал

Матье Жермен MATHIEU.GERMAIN2@USHERBROOKE.CA

Университет Шербрука, Канада

Кароль Грегор КАРОЛ.ГРЕГОР@GMAIL.COM

Google DeepMind

Иэн Мюррей I.MURRAY@ED.AC.UK

Эдинбургский университет, Соединенное Королевство

Хьюго Ларошель ГЮГО.ЛАРОШЕЛЬ@USHERBROOKE.CA

Университет Шербрука, Канада

Таблица 7. Отрицательное логарифмическое правдоподобие и 95% доверительные интервалы для Таблица 4 в основном документе.

Набор данных	СДЕЛАННЫЙ		Эонаде
	Фиксированная маска	Выборка по маске	16 орд.
Взрослые	13,12±0,05	13,13±0,05	13,19±0,04
Connect4	11,90±0,01	11,90±0,01	12,58±0,01
ДНК	83,63±0,52	79,66±0,63	82,31±0,46
грибы	9,68±0,04	9,69±0,03	9,69±0,03
NIPS-0-12	280,25±1,05	275,92±1,01	272,39±1,08
Ocr-буквы	28,34±0,22	30,04±0,22	27,32±0,19
RCV1	47,10±0,11	46,74±0,11	46,12±0,11
Интернет	28,53±0,20	28,25±0,20	27,87±0,20

Таблица 8. Бинаризованная отрицательная логарифмическая правдоподобие MNIST и 95% доверительные интервалы для таблицы 6 в основном документе.

Модель	
СДЕЛАНО 1гл (1 маска)	88,40±0,45
СДЕЛАНО 2гл (1 маска)	89,59±0,46
СДЕЛАНО 1гл (32 маски)	88,04±0,44
СДЕЛАНО 2гл (32 маски)	86,64±0,44