УДК 004.032.26

ЭФФЕКТИВНЫЙ АЛГОРИТМ ОБУЧЕНИЯ НА ОСНОВЕ СЛУЧАЙНОГО ПОИСКА

В.В. Мацкевич

Белорусский Государственный Университет,
пр. Независимости, 4, г. Минск, Беларусь Matskevich1997@gmail.com

В работе предлагается оригинальный распараллеленный алгоритм обучения нейронных сетей на основе метода отжига. Также предлагается архитектура нейронной сети ориентированной на параллельную обработку данных. Эффективность предложенного алгоритма обучения показана на примере решения задачи сжатия цветных изображений.

Ключевые слова: ограниченная машина Больцмана; параллельные вычисления; обучение; метод случайного поиска; метод отжига.

EFFECTIVE TRAINING ALGORITHM BASED ON RANDOM SEARCH

V.V. Matskevich

Belarusian State University,
Nezavisimosti av., 4, Minsk, Belarus Matskevich1997@gmail.com

In paper an original parallelized neural network training algorithm based on annealing method is proposed. Also, a neural network architecture focused on parallel data processing is proposed. Proposed training algorithm efficiency is shown on the example of solving color images compression problem.

Keywords: restricted Boltzmann machine; parallel computations; training; random search method; annealing method.

Введение

Современный мир вступил в эпоху цифровой экономики. В результате чего объемы информации поступающей из различных источников постоянно растут. Возникает необходимость в разработке эффективных алгоритмов, основанных на распараллеливании процесса обработки данных. В настоящее время большой популярностью пользуются нейросетевые технологии обработки данных. Однако для применения нейронных сетей необходимо произвести их обучение – трудоемкую настройку под решаемую задачу.

Для решения проблемы обучения были разработаны разнообразные алгоритмы на основе метода градиентного спуска. Они получили широкое

распространение за счет быстрой сходимости. Однако, данные алгоритмы имеют общую проблему – невысокое качество полученного решения что может быть неприемлемо в ряде случаев. Для решения данной проблемы существуют алгоритмы обучения основанные на идее случайного поиска. В данном случае пространство поиска оптимального решения существенно расширяется – это повышает качество решения. Одним из них является метод отжига. Он при определенных условиях сходится к оптимальному решению, причем из любого начального приближения [1]. Однако он обладает низкой скоростью сходимости, из-за чего считается не применимым на практике. На сегодняшний день вычислительных мощностей вполне достаточно для реализации данного метода.

В работе предлагается распараллеленный алгоритм обучения нейронных сетей на основе метода отжига. Предлагается специальная архитектура ограниченной машины Больцмана для параллельной обработки данных. Эффективность полученной системы демонстрируется на примере решения задачи сжатия цветных изображений.

1. Архитектура нейронной сети

Ограниченные машины Больцмана (ОМБ) применяются в задачах информационного поиска, анализа данных, сжатия изображений. ОМБ лежат в основе глубоких доверительных сетей. Несколько последовательно идущих ОМБ снижают размерность входных данных и делают их пригодными для последующей нейросетевой обработки.

Известно, что в основе ограниченной машины Больцмана лежит стохастический нейрон. Формально ее можно представить полносвязным двудольным графом $G=(X, U)$ [2],

$$\begin{cases} X = X_1 \cup X_2, X_1 \cap X_2 = \varnothing \\ U = \{u = (x_1, x_2) \mid \forall x_1 \in X_1, \forall x_2 \in X_2\} \end{cases},$$

где X – множество вершин – стохастических нейронов, U – множество ребер – синаптических связей, при этом вершины подмножества X_1 – задают нейроны входного слоя, а X_2 – нейроны выходного слоя.

Число нейронов во входном слое определяется размером входного образа, а количество нейронов в выходном слое определяется исходя из требований к степени сжатия данных.

Выходные сигналы слоев ОМБ реализуют некоторые законы вероятностного распределения. В зависимости от используемых законов распределения строят различные типы машин. В данной работе речь пойдет о

машинах типах Гаусс-Бернулли (ГБ) и Бернулли-Бернулли (ББ), т.к. они являются наиболее распространенными.

Для ОМБ типа ГБ каждой вершине входного слоя поставим в соответствие множества параметров $VB = \{b\}$ – смещения и $\sigma = \{\sigma\}$ – дисперсии вершин, а вершинам выходного слоя – множество параметров $HB = \{g\}$ – смещение вершин. Размеры множеств равны соответственно $|VB| = |\sigma| = |X_1|$, $|HB| = |X_2|$. Каждому ребру, связывающему пару вершин входного и выходного слоев поставим в соответствие множество параметров $W = \{w\}$ – весов ребер. Размер множества равен следующей величине $|W| = |X_1||X_2|$. Таким образом, описанное семейство нейронных сетей можно задать четырьмя типами параметров:RBM = (W, VB, σ, HB). Стоит отметить, что у ОМБ типа ББ отсутствует множество параметров σ.

Для обучения нейронной сети объем обучающей выборки должен быть не меньше числа настраиваемых параметров сети. Вычислительная сложность обучения прямопропорциональна произведению количества настраиваемых параметров сети на объем выборки. Количество настраиваемых параметров прямопропорционально размерности входных данных. Таким образом, вычислительная сложность обучения прямопропорциональна квадрату размерности входных данных.

Для решения проблемы вычислительной сложности обучения предлагается следующая архитектура ОМБ. Пусть входные данные имеют размерность N. Каждый элемент входных данных разбивается k равных фрагментов размера m ($km = N$). Значение k определяется при проектировке архитектуры нейронной сети. После разбиения данных создаются k ОМБ одинаковой архитектуры (по одной для каждого фрагмента данных). Таким образом исходные данные обрабатываются с помощью ансамбля из k ОМБ. Данный подход обладает следующими преимуществами: k ОМБ независимы друг от друга, что позволяет производить параллельное обучение машин; вычислительная сложность обучения прямопропорциональна размерности входных данных (линейная зависимость); снижение числа настраиваемых параметров нейронной сети понижает сложность обучения и повышает качество полученного решения (при разбиении данных на не слишком малые фрагменты).

2. Алгоритм обучения

Задача обучения в случае ее решения методами случайного поиска может быть сформулирована следующим образом. Пусть на конечном множестве допустимых решений Ω определена целевая функция F, и для каждого элемента $x \in \Omega$ задано множество соседних элементов

$N(x) \subset \Omega$. Задачу условной оптимизации в данном случае можно задать в виде тройки (Ω, F, N). Рассмотрим возможности ее решения с помощью случайного поиска, в частности, с помощью метода отжига.

Перед изложением алгоритма обучения, необходимо понять особенности нейронных сетей. Любая нейронная сеть состоит из одного или нескольких слоев. В зависимости от типа сети ее отдельные слои могут выполнять абсолютно разные преобразования входных данных для слоя. В зависимости от этого слой может состоять из различных фрагментов выполняющих абсолютно разные преобразования. Любой фрагмент нейронной сети задается набором параметров. Из-за того, что фрагменты выполняют разные преобразования то и соответствующие им наборы параметров имеют абсолютно разный диапазон значений. Таким образом, любая архитектура нейронной сети может быть задана объединением из m наборов параметров $NN = (x_1,x_2,\ldots,x_m)$. А конкретная сеть получается путем фиксации значений всех ее параметров.

Опишем теперь предлагаемый алгоритм, реализующий метод отжига.

Предварительный шаг. Инициализация начального состояния нейронной сети $NN_0 = (x_{10},x_{20},\ldots,x_{m0})$, величины T_0 и последовательности температур, связанных следующим соотношением:

$$T_k = T_0 / \ln(k+3), k>0$$

Общий *k*-ый шаг. Шаг 1. Генерация m равномерно распределенных дискретных случайных величин (СВ) $a_1,a_2,\ldots,a_m$ на отрезке от нуля до количества параметров в наборе. Генерация m случайных перестановок длиной, равной количеству параметров в наборе. Первые $a_1,a_2,\ldots,a_m$ элементов перестановок задают индексы изменяемых параметров в каждом наборе параметров соответственно.

Шаг 2. Генерация нового решения. Для каждого изменяемого параметра генерируется две равномерно распределенные СВ b, c на отрезках [0;1], [0;l/2]. Величина l зависит от того, какому набору принадлежит изменяемый параметр и равна l_1, l_2, …, l_m соответственно. Значения l для каждого набора задаются как параметры алгоритма. Пусть x_i – изменяемый параметр, а его новое значение x_i', тогда:

$$x'_i = \begin{cases} x_i + g, e \le 0{,}5 \\ x_i - g, e > 0{,}5 \end{cases}$$

Шаг 3. Принятие решения о переходе. Пусть x текущее решение, y – новое, сгенерированное на шаге 2 решение. Тогда решение x' на следующей итерации определяется следующим образом:

$$P(x'=y\,|\,x)=\min\left\{1,\exp\left((F(x)-F(y))/T_k\right)\right\}$$

Шаг 4. Проверка критерия остановки. Если время на обучение нейронной сети истекло, то алгоритм завершается. В противном случае производится переход на следующую итерацию.

Отдельная итерация алгоритма обучения на основе метода отжига, как было указано выше, состоит из 4 этапов. Все этапы кроме вычисления значения функционала для нового решения выполняет процессор. Наиболее трудоемким этапом является вычисление значения функционала и, при малой архитектуре сети, генерация нового решения. Вычислительная мощность видеокарты выше процессора в среднем в 20 раз. Это приводит к тому, что при обучении небольших нейронных сетей значительная часть времени расходуется на генерацию новых решений. Все этапы в отдельной итерации выполняются строго последовательно.

Алгоритм производит сокрытие вычислений. Рассмотрим подробнее этап вычисления значения функционала. Пусть текущее решение равно x, а новое решение – y. Во время вычисления значения функционала на видеокарте для решения y, процессор одновременно генерирует два новых решения $x_1 \in N(x)$ и $y_1 \in N(y)$. После вычисления значения функционала производится проверка необходимости перехода в новое решение. Если новое решение принято, то следующим решением, которое будет проверяться будет y_1 в противном случае x_1. Данный алгоритм позволяет замаскировать этап генерации нового решения. Таким образом, при переходе на следующую итерацию сразу будет производиться вычисление значения функционала для нового решения без его явной генерации.

Рассмотрим эффективность данного алгоритма. Пусть на генерацию одного нового решения на процессоре требуется a времени, вычисления значения функционала для нового решения на видеокарте b времени, а для передачи данных, синхронизации вычислений и прочего требуется c времени. Тогда в последовательном случае выполнения алгоритма обучения на одну итерацию требуется $a + b + c$ времени, т.к. остальные этапы пренебрежимо малы по объему вычислений. В случае применения алгоритма распараллеливания в процессе обучения генерируется сразу два новых решения и время одной итерации составит $\max(2a,b)+c$. Рассмотрим величину выигрыша времени r. Возможны два случая.

1. $2a \geq b$ в таком случае выигрыш составит $r = (a+b) - 2a = b - a$

2. $2a < b$ в таком случае получим $r = (a+b) - b = a$

Таким образом, алгоритм распараллеливания целесообразно применять, если время вычислений на видеокарте превышает время вычислений на процессоре. Максимальная эффективность достигается в случае, когда

величина *c* мала и видеокарта ровно в 2 раза быстрее процессора и время вычислений в таком случае сокращается на треть.

3. Результаты

Эффективность алгоритма обучения проверим на примере решения задачи сжатия цветных изображений выборки CIFAR-10 [3]. Для экспериментов выбрана 16-кратная степень сжатия. Более высокая степень приводит к большим потерям, более низкая – не имеет смысла, т.к. классические алгоритмы сжатия более эффективны в таких условиях. Сравнение проведено с методом адаптивного момента – наилучшего градиентного алгоритма обучения нейронных сетей [4] (см. Таблица)

Таблица – Результаты экспериментов

Алгоритм обучения	Метод отжига	Метод градиентного спуска
MSE	459	2940
PSNR	21,6	13,7
SSIM	0,749	0,227
время обучения, ч	1	1

Заключение

Разработанный алгоритм обучения нейронных сетей на основе случайного поиска заметно эффективнее широко распространенного градиентного. Полученный результат обладает перспективой. Он позволяет значительно увеличить эффективность обучения нейронных сетей. Благодаря широкой распространенности сетей возможно улучшение качества решений большого количества прикладных задач.

Библиографические ссылки

1. Hajek B. Cooling schedules for optimal annealing // Mathematics of operations research, vol. 13, iss. 2, 1988.
2. Krasnoproshin V.V., Matskevich V.V. Neural Network Data Processing Based on Deep Belief Networks // Communications in Computer and Information Science. Vol. 1282: Open Semantic Technologies for Intelligent System. Springer. 2020. P. 234–244.
3. Выборка CIFAR-10. URL: https://www.cs.toronto.edu/~kriz/cifar.html (дата доступа: 04.03.2020).
4. Kingma D.P., Ba J.L. Adam: A Method for Stochastic Optimization // Proceedings of the 3rd International Conference on Learning Representations. 2015. P. 1–15.