УДК 004.032.26

АЛГОРИТМЫ СЛУЧАЙНОГО ПОИСКА В ОБУЧЕНИИ НЕЙРОННЫХ СЕТЕЙ

В.В. МАЦКЕВИЧ

Белорусский государственный университет, Республика Беларусь

Поступила в редакцию 20 марта 2023

Аннотация. В работе рассматривается проблема обучения нейронных сетей. Предложены алгоритмы обучения на основе метода отжига и генетического алгоритма. Показано, что разработанные алгоритмы обладают большей эффективностью, чем существующие градиентные методы.

Ключевые слова: метод отжига, генетический алгоритм, метод градиентного спуска, обучение, нейронная сеть.

Введение

В настоящее время происходит стремительное развитие цифровых технологий. Объемы получаемых данных из различных источников стремительно растут [1]. Возникает проблема автоматизации эффективного извлечения полезной информации из больших данных.

Для решения данной проблемы был разработан нейросетевой подход. Проектируется архитектура нейронной сети, настраивается под конкретную предметную область и задачу и применяется для решения задачи. Данный подход является наиболее перспективным из-за универсальности применения нейронных сетей. На практике они способны адаптироваться под любую прикладную задачу, что делает их универсальными. Процесс настройки нейронной сети под прикладную задачу называется обучением. Эффективность полученного решения напрямую зависит от результата обучения [2].

Из-за постоянного роста объема обрабатываемых данных (количества и размерности) постоянно растет размер нейронной сети. Это делает обучение наиболее трудоемким этапом нейросетевой обработки.

Несмотря на достигнутые в данном направлении результаты, проблема обучения по-прежнему является актуальной.

Существуют два основных подхода к обучению нейронных сетей. Один базируется на направленном поиске (перемещение по множеству решений по строго определенному правилу), а другой – на случайном поиске. Наиболее популярными являются градиентные методы (направленный поиск), которые на практике обучают нейронные сети за приемлемое время. Представители другого подхода – методы отжига и генетические алгоритмы, обеспечивают хорошее качество, но работают существенно медленнее.

В работе сравнивается эффективность метода отжига, генетического алгоритма и градиентного спуска для обучения нейронной сети на примере решения задачи сжатия цветных изображений.

Анализ проблемы

Обучение нейронной сети можно сформулировать в виде следующей задачи.

Пусть задана некоторая нейронная сеть N и обучающая выборка, описывающая входные данные задачи X. Пусть задан некоторый функционал качества f описывающий качество полученного решения. В качестве аргументов данная функция принимает нейронную сеть N и выборку X. Данный функционал, как и обучающая выборка, определяются решаемой

прикладной задачей. Необходимо настроить значения параметров сети X таким образом, чтобы достичь минимума функционала f.

Таким образом, обучение нейронной сети является оптимизационной задачей и для нее решения можно использовать все оптимизационные методы.

Наибольшей перспективой обладают приближенные итерационные методы оптимизации. Они универсальны, позволяют регулировать точность полученного решения в зависимости от ограничений на вычислительные ресурсы.

Все итерационные методы оптимизации разделяются на два класса: направленные и ненаправленные методы.

Направленные методы характеризуются наличием строгого правила перехода из рассматриваемого решения в новое. К направленным методам относятся все методы на основе вычисления градиента: метод простого градиента, метод моментов (тяжелого шарика), градиент Нестерова, метод адаптивного момента и т.п. На ранних этапах развития нейронных сетей вычислительные мощности компьютеров были крайне малы. Для обучения сетей был необходим алгоритм с быстрой сходимостью и малым объемом вычислений на отдельной итерации. Любой направленный метод за счет наличия правила переходов существенно ограничивает множество рассматриваемых решений и обладает высокой скоростью сходимости, однако, это приводит к тому, что может быть упущено оптимальное решение. Градиентные методы помимо высокой скорости сходимости, на практике способны получать приемлемое по качеству решение, что обусловило их широкое распространение на начальной стадии развития нейронных сетей и главными методами обучения по сей день. Однако, градиентные методы требуют дифференцируемости целевой функции и могут останавливаться в решениях с около нулевым значением градиента, что приводит к необходимости наложения ограничений на архитектуры нейронных сетей и не всегда близкому к оптимальному решению.

Альтернативным подходом являются ненаправленные методы. В ненаправленных методах на каждой итерации новое решение выбирается случайным образом на основе текущего решения. К алгоритмам случайного поиска относятся генетический алгоритм и алгоритмы отжига. В первом случае на каждой итерации рассматривается конечное множество решений и из него случайным образом генерируется новое множество решений. Во втором случае на каждой итерации рассматривается одно решение и из него случайным образом формируется новое решение. Алгоритмы случайного поиска (ненаправленные методы) не ограничивают множество рассматриваемых решений, что теоретически позволяет достичь оптимального решения, однако из-за того, что множество рассматриваемых решений на практике весьма велико, требуется большое количество итераций для получения решения. Долгое время считалось, что алгоритмы случайного поиска не применимы на практике из-за неприемлемо высокой вычислительной сложности, однако благодаря быстрому росту вычислительных мощностей они способны за приемлемое время получить решение.

Для сравнения двух принципиально различных подходов к решению оптимизационных задач, рассмотрим наиболее сильных представителей из каждого класса. Из направленных методов – метод адаптивного момента, из ненаправленных – метод отжига и генетический алгоритм.

Алгоритм обучения на основе метода отжига

Пусть на конечном множестве допустимых решений Ω определена целевая функция F, и для каждого элемента x множества задано множество соседних элементов $N(x)$. Задачу условной оптимизации в данном случае можно задать в виде тройки (Ω, F, N). Рассмотрим возможности ее решения с помощью метода отжига. Алгоритм включает следующие основные шаги.

Шаг 1. *Предварительный этап*. Инициализация начального состояния нейронной сети $Net_0 = Net(x_{10}, x_{20}, \ldots, x_{m0})$ и последовательности температур $T_0, T_1, \ldots, T_k$, связанных соотношением: $T_k = \dfrac{T_0}{\ln(k+2)}$, $k > 0$, где T_0 – заранее заданное значение.

Шаг 2. *Общая k-я итерация*.

2.1. *Генерация случайных величин*. Генерируется m равномерно распределенных на отрезке от нуля до количества параметров в наборе дискретных случайных величин $a_1, a_2, \ldots, a_m$. Генерируется m случайных перестановок длиной, равной количеству в наборе параметров. Первые $a_1, a_2, \ldots, a_m$ элементов каждой перестановки задают индексы изменяемых параметров в каждом наборе параметров соответственно.

2.2. *Генерация нового решения.* Для каждого изменяемого параметра генерируется равномерно распределенная на отрезке длины l с центром в нуле случайная величина b. Величина l зависит от того, какому набору принадлежит изменяемый параметр и равна $l_1, l_2, \ldots, l_m$ соответственно. Значения l для каждого набора задаются как параметры алгоритма.

Пусть x_i – изменяемый параметр, а его новое значение x_i', тогда: $x_i' = x_i + b$.

2.3. *Принцип перехода.* Пусть x текущее решение, y – новое, сгенерированное на шаге 2 решение. Тогда решение x' на следующей итерации определяется следующим образом: $P(x' = y \mid x) = \min\left\{1, \exp\left(\frac{F(x) - F(y)}{T_k}\right)\right\}$.

2.4. *Критерий останова.* Если время на обучение нейронной сети истекло, то алгоритм завершается. В противном случае производится переход на следующую итерацию.

Как было показано ранее, что предложенный алгоритм обучения на основе метода отжига сходится по вероятности к оптимальному решению, причем из любого начального приближения [3]. Однако, как и большинство алгоритмов случайного поиска, он обладает низкой скоростью сходимости. В экспериментах будет проверена эффективность предложенного алгоритма.

Алгоритм обучения на основе генетического алгоритма

Генетический алгоритм является эвристикой, реализующей идею жадного поиска. Данный алгоритм в зависимости от реализации может быть, как бесполезным, так и крайне эффективным при решении прикладных задач. Однако его главный недостаток – крайне медленная скорость сходимости. Более того, данный алгоритм не гарантирует сходимости к оптимальному решению.

Для проведения экспериментов был разработан следующий вариант генетического алгоритма. Предполагается, что целевую функцию необходимо минимизировать.

Шаг 1. *Предварительный этап*. Генерация нескольких случайных решений. Каждое отдельное решение является полноценной нейронной сетью, архитектура которой задается перед запуском алгоритма обучения и не изменяется. Количество решений N – параметр алгоритма. Для каждого решения вычисляется значение целевой функции.

Шаг 2. *Общая k-я итерация*.

2.1. *«Мутация»*. Из текущего множества решений выбирается решение с наибольшим значением целевой функции. Для выбранного решения производится "мутация" – генерация нового решения из текущего по той же схеме, что и для разработанного алгоритма отжига. Единственное отличие – значения параметров отжига могут отличаться от генетического алгоритма.

2.2. *«Селекция»*. Для полученного решения вычисляется значение целевой функции. Если значение целевой функции для нового решения меньше чем для текущего, то производится замена текущего решения на новое, в противном случае новое решение отбрасывается.

2.3. *«Скрещивание»*. Случайным образом из множества текущих решений выбирается два решения – *a, b*. Для всех значений параметров решений a, b производятся следующие расчеты:

$$\begin{cases} d_i = b_i - a_i \\ c_i = a_i + d_i \cdot \alpha \\ \alpha \in [0; \varphi], 0 < \varphi \le 1 \end{cases},$$

где α – равномерно распределенная случайная величина на отрезке, где φ – параметр алгоритма.

2.4. *«Отбор»*. Для полученного решения вычисляется значение целевой функции. Во множестве решений выбирается решение с наибольшим значением целевой функции. Если значение целевой функции у нового решения меньше, то старое решение заменяется на новое решение, в противном случае новое решение отбрасывается.

2.5. *Критерий останова*. Если время, отведенное на обучение, истекло, алгоритм завершает свою работу, в противном случае производится переход на следующую итерацию.

Из недостатков данного алгоритма также стоит отметить, что, как и любая реализация генетического алгоритма требуется «хорошая» в некотором смысле генерация случайных решений на предварительном шаге. В противном случае скрещивание утрачивает смысл, и алгоритм вырождается в жадный поиск – из одного единственного решения генерируется новое случайное решение путем «мутации» и производится выбор лучшего решения.

Метод градиентного спуска

Метод градиентного спуска заключается в вычислении на каждой итерации градиента целевой функции и изменения значений оптимизируемых параметров в направлении, противоположном градиенту. Методы градиентного спуска обладают высокой скоростью сходимости (не ниже линейной). Например, метод секущих имеет скорость сходимости выше линейной, но ниже квадратической, метод Ньютона имеет квадратическую скорость сходимости. Однако, любой градиентный метод имеет проблему остановки в решениях с околонулевыми значениями градиента. Это могут быть окрестности точек перегиба, локальных минимумов, не совпадающих с глобальным и т.п.

Для решения данной проблемы были разработаны модификации метода: метод моментов (тяжелого шарика), градиент Нестерова метод адаптивного момента [4], метод следования за движущимся лидером [5]. Все модификации градиента заключаются в специфическом правиле пересчета оптимизируемых параметров на основе вычисленных градиентов на различных итерациях. Однако, градиент накладывает ограничения на архитектуру нейронных сетей для решения проблемы затухающего и взрывного градиента и не гарантируют сходимость к оптимальному решению.

В экспериментах для обучения нейронных сетей от направленных методов оптимизации будет использован метод адаптивного момента. Он является одним из наиболее эффективных и обладает стабильностью в качестве полученного решения в отличие от метода следования за движущимся лидером.

Эксперименты

Для сравнения эффективности алгоритмов обучения нейронных сетей на основе различных методов будем решать задачу сжатия изображений на выборке CIFAR-10 [6].

Для экспериментов были выбраны 8-кратное, 16-кратное и 32-кратное сжатия. Более низкие степени сжатия эффективнее производятся с помощью классических алгоритмов сжатия, а более высокие не имеют смысла из-за слишком больших потерь.

Для всех степеней сжатия изображения были разбиты на фрагменты по 4×4 пикселя. Разбиение на меньшие фрагменты приводит к снижению качества сжатия, увеличение, в свою очередь, приводит к слишком большой архитектуре нейронной сети и требует слишком большого объема данных и вычислительных ресурсов для обучения. Каждый отдельный фрагмент сжимает отдельная ограниченная машина Больцмана типа Гаусс-Бернулли. Для 8-кратного сжатия количество нейронов в скрытом слое каждой машины было 48, для 16-кратного – 24, для 32-кратного – 24, но для достижения необходимой степени сжатия был добавлен еще один слой из ограниченных машин Больцмана типа Бернулли-Бернулли с 48 нейронами во входном слое и 24 – в скрытом.

Для обучения ограниченных машин Больцмана градиентным методом будет использоваться алгоритм CD-1. Алгоритм PCD на небольшом числе итераций более эффективный [7], однако, он строится на предположении, что на отдельной итерации параметры обучаемой сети изменяются не существенно, что не соответствует решаемой задаче. Алгоритм CD-k требует в k раз большего объема вычислений на отдельной итерации, чем CD-1 и при этом

достигает лучшего качества [8], однако в решаемой задаче значения градиентов очень велики и использование алгоритма CD-k не целесообразно.

Для обучения первые 8000 изображений были использованы в качестве обучающей выборки, последующие 7000 изображений для валидационной, остальные 45000 изображений сформировали тестовую выборку.

Для оценки эффективности сжатия были выбраны наиболее распространенные функционалы качества MSE (среднеквадратическая ошибка), PSNR (соотношение максимального сигнала к помехе), PSNR-HVS (PSNR с поправкой на особенности человеческого зрения), SSIM (структурная схожесть изображений).

Эксперименты проводились на операционной системе Lubuntu 20.04 с процессором intel i7-4770k, 16 GB 1600MHz оперативной памятью и видеокартой nvidia rtx 3070. Все алгоритмы были реализованы в рамках фреймворка с помощью библиотек OpenCL и OpenMP на языке C++. Фреймворк был настроен на использование видеокарты для обучения.

Время обучения нейронных сетей выбиралось индивидуально для обеспечения наилучшего соотношения качество обучения к затраченному времени (табл. 1).

Табл. 1. **Результат обучения ограниченных машин Больцмана**

Алгоритм обучения	Метод адаптивного момента			Генетический алгоритм			Метод отжига		
Степень сжатия, бит/пиксель	3	1,5	0,75	3	1,5	0,75	3	1,5	0,75
MSE	270	435	917	349	466	798	253	420	670
PSNR	23,9	21,9	18,6	22,8	21,6	19,2	24,2	22,0	20,0
PSNR_HVS	24,2	22,1	18,8	23,0	21,8	19,3	24,3	22,2	20,1
SSIM	0,834	0,765	0,600	0,794	0,747	0,629	0,837	0,762	0,674
Время обучения, ч	1,4	0,52	0,78	2	2	3	4	3	3

Из результатов экспериментов видно, что метод отжига превосходит остальные алгоритмы по качеству полученного решения, однако он примерно втрое медленнее градиентного алгоритма. Обучение нейронных сетей при решении прикладных задач осуществляется лишь однажды, следовательно, трехкратное отставание отжига от градиента по времени обучения не является критическим и является приемлемым.

Генетический алгоритм хоть и превосходит по качеству градиент в сложном случае (обучение ограниченной машины Больцмана типа Бернулли-Бернулли), но уступает методу отжига. Он достигает примерно равного качества с отжигом за счет увеличения времени обучения в десять раз – но это неприемлемо много.

Для быстрого обучения лучшего всего подходит градиентный алгоритм, однако качество полученного решения в таком случае не гарантируется. Более того, по мере роста времени обучения генетический алгоритм и метод отжига продолжают повышать качество полученного решения, в то время как градиент не изменяет качество полученного решения.

Заключение

Экспериментально было показано, что метод градиентного спуска уступает по качеству полученного решения генетическому алгоритму и методу отжига при решении сложных оптимизационных задач. Также экспериментально было показано, что предложенный алгоритм на основе метода отжига является наиболее эффективным алгоритмом обучения, в то время как эффективность генетического алгоритма зависит от конкретной его реализации.

По мере роста вычислительных мощностей компьютеров, метод отжига и генетический алгоритм совершают большее число итераций за фиксированное время, отведенное на обучение, и тем самым повышается качество полученного решения. Для градиентных методов при росте мощностей снижается лишь время обучения, но качество остается неизменным.

Из этого можно сделать вывод, что алгоритмы случайного поиска обладают определенным потенциалом в решении задачи обучения нейронных сетей.

RANDOM SEARCH ALGORITHMS IN NEURAL NETWORKS TRAINING

V.V. MATSKEVICH

Abstract. The paper deals with a neural network training problem. Training algorithms based on the annealing method and the genetic algorithm are proposed. It is shown that the developed algorithms are more efficient than the existing gradient methods.

Keywords: annealing method, genetic algorithm, gradient descent method, training, neural network.

Список литературы

1. Choi Y., El-Khamy M., Lee J. IEEE/CVF International Conference on Computer Vision (ICCV), 2019. P. 3146–3154.
2. You Ya., Li J., Reddi S., [et al]. ICLR, 2020. P. 1–37.
3. Krasnoproshin V.V., Matskevich V.V. // Proc. of the 13-th International Conference "Computer Data Analysis and Modeling". 2022. P. 96–99.
4. Kingma D.P., Ba J.L. // Proc. of the 3rd International Conference on Learning Representations. 2015. P. 1–15.
5. Zheng Sh., Kwok J.T. Proc. of the 34th International Conference on Machine Learning, 2017. Vol. 70. P. 4110–4119.
6. Выборка CIFAR-10 [Электронный ресурс]. URL: https://www.cs.toronto.edu/~kriz/cifar.html.
7. Oswin K., Fischer A., Igel Ch. // Pattern Recognition Letters. 2018. Vol. 102. P. 1–7.
8. Li X., Gao X., Wang Ch. // IEEEACCESS. 2021. Vol. 9. P. 21939–21950.