

Тема 7

Градиентные методы обучения нейронных сетей

Equation Chapter (Next) Section 7

В предыдущей лекции был рассмотрен алгоритм обучения методом обратного распространения ошибки. Этот алгоритм является исторически одним из первых градиентных алгоритмов обучения НС. В текущей лекции будет рассмотрен ряд других эффективных градиентных алгоритмов обучения НС.

Общая постановка задачи обучения НС с нелинейными элементами эквивалентна задаче построения приближения (аппроксимации) некоторой функции $\bar{y} = r(\bar{x})$, где r - нелинейная функция аппроксимируемая некоторой нейронной сетью. При условии априорности и неизменности архитектуры этой НС, задача обучения сводится к поиску некоторого вектора параметров $\bar{w}$, для которого функция $\bar{y} = f(\bar{w}, \bar{x})$ наилучшим образом приближает значения искомой функции r .

Постановка задачи

Для некоторой функции $r(x)$, заданной обучающей выборкой $(\bar{x}^i, \bar{d}^i)$, $i = 1, \dots, N$, необходимо найти вектор параметров $\bar{w}'$ такой, что НС реализующая функцию $\bar{y} = f(\bar{w}', \bar{x})$ наилучшим образом аппроксимирует функцию r , т. е. верно

$$D_r(f(\bar{w}', \bar{x})) = \sum_{n=1}^N e_n = \min_{\bar{w}} D_r(f(\bar{w}, \bar{x})), \quad (7.1)$$

где e_n - ошибка НС на n -й паре выборки.

Как правило, для вычисления ошибки на одной паре применяют формулу

$$e_n = \frac{1}{2} \|\bar{y}^n - f(\bar{w}, \bar{x}^n)\|^2. \quad (7.2)$$

В этом случае, функционал D_r будем обозначать $D_r(f(\bar{w}, \bar{x})) \equiv E(\bar{w})$, и формула его вычисления примет вид

$$E(\bar{w}_k) = \frac{1}{2} \sum_{n=1}^N (\bar{y}^n - f(\bar{w}_k, \bar{x}^n))^2. \quad (7.3)$$

Большинство градиентных методов первого порядка опираются на вычисления по формуле

$$\bar{w}_{k+1} = \bar{w}_k + h_k \bar{p}_k, \quad (7.4)$$

где $\bar{p}_k$ задаёт направление, а h_k - размер шага.

Обобщённый алгоритм нелинейной оптимизации состоит из следующих шагов [1].

1. Выбираем начальное направление движения $\bar{p}_0$ (например, случайным образом) и начальный шаг $h_0 \approx 1$.
2. На очередном шаге определяем направление движения $\bar{p}_k$, основываясь на требуемой для этого информации – например, векторе $\bar{p}_{k-1}$ или градиенте функционала ошибки в точке.
3. Определяем очередной шаг h_k , обычно определяя минимум функционала ошибки в направлении $\bar{p}_k$ с некоторой точностью.
4. Вычисляем новое значение $\bar{w}_k$ по формуле (7.4).
5. Если ошибка, вычисляемая по формуле (7.3) недостаточно мала, и не выполнено какое-либо правило останова, то переход на шаг 2, иначе – конец.

Таким образом, конкретизация параметров h_k и $\bar{p}_k$ даёт нам некоторый конкретный алгоритм. При этом, в зависимости от способа вычисления параметра $\bar{p}_k$ методы делятся на методы нулевого порядка – те которые не требуют вычисления градиента функционала ошибки, методы первого порядка – требующие вычисления первых частных производных ФО и методы второго порядка – требующие вычисления оценки матрицы частных производных второго порядка и обратной к ней. Использование частных производных более высоких порядков для вычисления направления движения оказалось крайне неэффективным.

Методы первого порядка

Рассмотрим некоторые способы вычисления величина шага и направления.

Метод наискорейшего спуска

Основные шаги алгоритма следующие.

1. В качестве начального значения p_0 выбирается некоторый случайный вектор. В качестве h_0 , обычно берётся число близкое к 0,1.
2. На втором шаге $\bar{p}_k = \bar{g}_k \equiv -\nabla E(\bar{w}_k)$.
3. Шаг h_k определяется из условия минимума функции $E(\bar{w}_k + h_k \bar{p}_k)$.
4. Ошибка считается в соответствии с формулой (7.3) и осуществляется либо переход на шаг 2, либо остановка алгоритма.

Метод тяжёлого шарика

Направление задается следующим образом

$$\bar{p}_k = \bar{g}_k + b \bar{p}_{k-1}; \bar{p}_0 = \bar{g}_0 = -\nabla E(\bar{w}_0), \quad (7.5)$$

где b - некоторая константа из интервала $[0,1]$, задаваемая пользователем.

В этом алгоритме, в отличие от алгоритма наискорейшего спуска, скорость сходимости намного быстрее, так как даже при резко меняющемся векторе направления, процесс имеет инерционность.

Методы сопряженных градиентов

Если отказаться от постоянности коэффициента b , то из метода тяжёлого шарика можно получить один из методов сопряженных градиентов. Приведем параметры этих алгоритмов в виде таблицы.

Таблица 7.1

Параметры алгоритма		Название
$\bar{p}_k = \bar{g}_k + b_k \bar{p}_{k-1};$ $\bar{p}_0 = \bar{g}_0 = -\nabla E(\bar{w}_0)$	$b = const \in [0,1]$	метод тяжёлого шарика
	$b_k = \frac{(\bar{g}_k, \bar{g}_k)}{(\bar{g}_{k-1}, \bar{g}_{k-1})}$	метод Флетчера-Ривса
	$b_k = \frac{(\bar{g}_k - \bar{g}_{k-1}, \bar{g}_k)}{(\bar{g}_k - \bar{g}_{k-1}, \bar{p}_k)}$	метод Хестенса-Штифеля
	$b_k = \frac{(\bar{g}_k - \bar{g}_{k-1}, \bar{g}_k)}{(\bar{g}_{k-1}, \bar{g}_{k-1})}$	метод Полака-Рибьера
	$b_k = \frac{(\bar{g}_k - \bar{g}_{k-1}, \bar{g}_k)}{(\bar{g}_{k-1}, \bar{p}_{k-1})}$	без названия
	$b_k = \frac{(\bar{g}_k, \bar{g}_k)}{(\bar{g}_{k-1}, \bar{p}_{k-1})}$	без названия
	$b_k = \begin{cases} gb_{k-1}, & (\bar{g}_k, \bar{p}_{k-1}) \geq 0; \\ \min \left(b_{k-1}, -\frac{1}{2} \frac{(\bar{g}_k, \bar{g}_k)}{(\bar{g}_k, \bar{p}_{k-1})} \right), & (\bar{g}_k, \bar{p}_{k-1}) < 0 \end{cases}$ $g > 1.$	Метод тяжёлого шарика регулируемой массы
$\bar{p}_0 = \bar{g}_0 = -\nabla E(\bar{w}_0),$ $\bar{p}_k = \bar{g}_k + b_k \bar{p}_{k-1} - g_k \bar{q}_k,$ $\bar{q}_k = \bar{g}_k - \bar{g}_{k-1}.$	$b_k = \frac{(\bar{q}_k, \bar{g}_k) + (\bar{g}_k, \bar{p}_{k-1}) \left(h_{k-1} + \frac{(\bar{q}_k, \bar{q}_k)}{(\bar{q}_k, \bar{p}_{k-1})} \right)}{(\bar{q}_k, \bar{p}_{k-1})},$ $g_k = \frac{(\bar{g}_k, \bar{p}_{k-1})}{(\bar{q}_k, \bar{p}_{k-1})}$	BFGS с конечной памятью

Алгоритм RProp.

Обозначим $\Delta w_i(k) = w_i(k) - w_i(k-1)$.

1. Устанавливаем начальные значения $\bar{w}_0, \bar{p}_0$.
2. Первый шаг делается в направлении антиградиента $\Delta w_i(k) = l g_i(0)$, $l = 1/N$.
3. На очередном шаге переменные пересчитываются в соответствии с формулой

$$\Delta w_i(k+1) = \begin{cases} g_1 \Delta w_i(k), & g_i(k)g_i(k-1) > 0, \\ g_2 \Delta w_i(k), & g_i(k)g_i(k-1) < 0, \\ \Delta w_i(k), & g_i(k)g_i(k-1) = 0. \end{cases} \quad (7.6)$$

где $g_1 > 1, g_2 < 1$ (например, 1,2 и 0,5).

4. Проверяется ошибка и в случае необходимости осуществляется переход на 3-й шаг.

Смысл метода – использовать только знак координат градиента, что позволяет стабилизировать скорость движения, что позволяет не оставаться в неглубоких локальных минимумах.

Алгоритм QuikProp

1. Устанавливаем начальные значения $\bar{w}_0, \bar{p}_0$.
2. Первый шаг делается в направлении антиградиента $\Delta w_i(k) = l g_i(0)$, $l = 1/N$.
3. На очередном шаге переменные пересчитываются в соответствии с формулой

$$\Delta w_i(k+1) = \begin{cases} b_k \Delta w_i(k), & b_k \leq b, \\ \tilde{b}_k \Delta w_i(k), & b_k > b. \end{cases} \quad (7.7)$$

$$b_k = \frac{g_i(k)}{g_i(k) - g_i(k-1)}, \quad \tilde{b}_k = b \cdot \text{sign}(b_k).$$

где b - некоторое число, заданное пользователем.

4. Проверяется ошибка и в случае необходимости осуществляется переход на 3-й шаг.

Алгоритм Delta-delta

1. Устанавливаем начальные значения $\bar{w}_0, \bar{p}_0$.
2. Первый шаг делается в направлении антиградиента $\Delta w_i(k) = l g_i(0)$, $l = 1/N$.
3. На очередном шаге переменные пересчитываются в соответствии с формулой

$$\begin{aligned} \Delta w_i(k+1) &= g_i(k)g_i(k), \\ g_i(k+1) &= g_i(k) + d g_i(k)g_i(k+1), \\ d &= d_0 / g_i^2(0), g_i(0) = g_0 / |g_i(0)|. \end{aligned} \quad (7.8)$$

Здесь константы d_0, g_0 задаются пользователем.

4. Проверяется ошибка и в случае необходимости осуществляется переход на 3-й шаг.

Методы второго порядка

Метод Ньютона

Пусть нам надо решить уравнение $\nabla E(\bar{\mathbf{w}}) = 0$. Если мы имеем некоторое приближение к решению $\bar{\mathbf{w}}_k$, тогда в окрестности этой точки

$$\nabla E(\bar{\mathbf{w}}) = \nabla E(\bar{\mathbf{w}}_k + \bar{\mathbf{w}} - \bar{\mathbf{w}}_k) \approx \nabla E(\bar{\mathbf{w}}_k) + \nabla^2 E(\bar{\mathbf{w}}_k)(\bar{\mathbf{w}} - \bar{\mathbf{w}}_k). \quad (7.9)$$

Будем искать $\bar{\mathbf{w}}_{k+1}$ следующим образом

$$\bar{\mathbf{w}}_{k+1} = \bar{\mathbf{w}}_k - \nabla^2 E^{-1}(\bar{\mathbf{w}}_k) \nabla E(\bar{\mathbf{w}}_k) = \bar{\mathbf{w}}_k + \nabla^2 E^{-1}(\bar{\mathbf{w}}_k) \mathbf{g}_k. \quad (7.10)$$

Из этого равенства следует, что E должна быть дважды дифференцируема по всем компонентам $\bar{\mathbf{w}}$, а матрица вторых производных $\nabla^2 E(\bar{\mathbf{w}}_k)$ положительно определена и обратима. Таким образом, для вычисления нового значения $\bar{\mathbf{w}}_{k+1}$ требуется рассчитать градиент, матрицу вторых производных и обратить её. Это довольно затратные операции, поэтому на практике, как правило, используя либо приближенное значение $\nabla^2 E^{-1}(\bar{\mathbf{w}}_k)$, либо рассчитывают его итеративно. В общем виде, такие схемы требуют расчёта направления движения по формуле

$$\bar{\mathbf{p}}_k = H_k \bar{\mathbf{g}}_k, \quad (7.11)$$

где H^k - некоторая матрица, заменяющая $\nabla^2 E^{-1}(\bar{\mathbf{w}}_k)$, т. е. матрица заменяющая обратную к гессиану.

Рассмотрим некоторые конкретные алгоритмы, так как они приведены в [1].

Алгоритм сопряженных направлений

1. Выбираем начальные значения подбираемых переменных $\bar{\mathbf{w}}_0$ и параметры алгоритма.
2. Вычисляем антиградиент по формуле $\bar{\mathbf{g}}_k = -\nabla E(\bar{\mathbf{w}}_k)$.
3. Вычисляем матрицу по формуле

$$H_{k+1} = H_k - \frac{\bar{\mathbf{z}}_k \bar{\mathbf{z}}_k^T}{(\bar{\mathbf{z}}_k, \bar{\mathbf{g}}_k - \bar{\mathbf{g}}_{k+1})}, \quad (7.12)$$

$$\bar{\mathbf{z}}_{k+1} = H_k (\bar{\mathbf{g}}_k - \bar{\mathbf{g}}_{k+1}). \quad (7.13)$$

4. Вычисляем направление по формуле $\bar{\mathbf{p}}_k = H_k \bar{\mathbf{g}}_k$.
5. Вычисляем шаг h_k каким либо методом одномерной минимизации.
6. Вычисляем ошибку E на обучающей выборке и если ошибка существенна, повторяем шаги 2-5.

Алгоритм BFGS

Этот алгоритм полностью повторяет алгоритм сопряженных направлений и отличается от него лишь способом вычисления значения на 3-м шаге:

$$H_{k+1} = H_k - \frac{1}{(\bar{p}_k, \bar{g}_k - \bar{g}_{k+1})} \cdot \left(\left(h_k + \frac{(\bar{z}_k, \bar{g}_k - \bar{g}_{k+1})}{(\bar{p}_k, \bar{g}_k - \bar{g}_{k+1})} \right) \bar{p}_k \bar{p}_k^T - \bar{z}_k \bar{p}_k^T - \bar{p}_k \bar{z}_k^T \right). \quad (7.14)$$

Алгоритм Левенберга-Марквардта

Для описания этого метода напомним представление функции E :

$$E(\bar{w}) = \frac{1}{2} \sum_{n=1}^M (\bar{y}^n - f(\bar{w}, \bar{x}^n))^2 \equiv \frac{1}{2} \sum_{n=1}^M e_n^2(\bar{w}). \quad (7.15)$$

Обозначим

$$\bar{e}(\bar{w}) = (e_1(\bar{w}), e_2(\bar{w}), \dots, e_N(\bar{w})), \quad (7.16)$$

$$J(\bar{w}) = \begin{pmatrix} \frac{\partial e_1}{\partial w_1} & \frac{\partial e_1}{\partial w_2} & \dots & \frac{\partial e_1}{\partial w_n} \\ \frac{\partial e_2}{\partial w_1} & \frac{\partial e_2}{\partial w_2} & \dots & \frac{\partial e_2}{\partial w_n} \\ \dots & \dots & \dots & \dots \\ \frac{\partial e_N}{\partial w_1} & \frac{\partial e_N}{\partial w_2} & \dots & \frac{\partial e_N}{\partial w_n} \end{pmatrix}. \quad (7.17)$$

В этих обозначениях шаги алгоритма выглядят следующим образом.

1. Выбираем начальные значения подбираемых переменных $\bar{w}_0$, $r > 1$, $v_0 \square 1$.

2. Вычисляем антиградиент по формуле

$$\bar{g}_k = \bar{e}(\bar{w}_k) J(\bar{w}_k). \quad (7.18)$$

3. Вычисляем матрицу $G(\bar{w}_k)$ по формуле

$$G(\bar{w}_k) = J(\bar{w}_k) J^T(\bar{w}_k) + v_k \cdot I. \quad (7.19)$$

4. Вычисляем направление по формуле

$$\bar{p}_k = -G^{-1}(\bar{w}_k) \bar{g}_k. \quad (7.20)$$

5. Вычисляем значение

$$v_k = \begin{cases} \frac{v_k - 1}{r}, & E\left(\frac{v_{k-1}}{r}\right) \leq E_k; \\ v_k = v_{k-1}, & E\left(\frac{v_{k-1}}{r}\right) > E_k \text{ и } E(v_{k-1}) < E_k; \\ v_k = v_{k-1} r^m, & E\left(\frac{v_{k-1}}{r}\right) > E_k \text{ и } E(v_{k-1}) > E_k. \end{cases} \quad (7.21)$$

6. Вычисляем ошибку E на обучающей выборке и если ошибка существенна, повторяем шаги 2-6.

В этом алгоритме $v_k \cdot I$ называется параметром Левенберга-Марквардта, является скалярной величиной, изменяющейся в процессе оптимизации. В начале процесс, когда $\bar{w}_k$ далеко от искомого решения, значение v_k значительно превосходит собственное значение матрицы $J(\bar{w}_k)J^T(\bar{w}_k)$, следовательно

$$G(\bar{w}_k) \cong v_k \cdot I, \quad (7.22)$$

и

$$p_k = -\frac{\bar{g}_k}{v_k}. \quad (7.23)$$

По мере уменьшения погрешности и приближения к искомому решению величина параметра v_k понижается, и первое слагаемое в формуле (7.19) начинает играть всё более важную роль.

Литература

1. Д. А. Тархов. Нейронные Сети. Модели и алгоритмы. Кн. 18. – М.: Радиотехника, 2005.