Математические основы теории искусственных нейронных сетей

Кафедра технологий программирования
Доцент Еникеева З.А.

Алгоритм 1.1. Обучение персептрона Розенблатта

Вход:

X^ℓ — обучающая выборка;

η — темп обучения;

Выход:

синаптические веса $w_0, w_1, \ldots, w_n$;

1: инициализировать веса w_j;
2: **повторять**
3: **для всех** $i = 1, \ldots, \ell$
4: $w := w - \eta\big(a(x_i) - y_i\big)x_i$;
5: **пока** веса w изменяются;

Правило Хэбба. Иногда удобнее полагать, что классы помечены числами -1 и 1, а нейрон выдаёт знак скалярного произведения:

$$a(x) = \text{sign}(\langle w, x \rangle).$$

Тогда несовпадение знаков $\langle w, x_i \rangle$ и y_i означает, что нейрон ошибается на объекте x_i. При этом выражение (1.2) преобразуется в следующее правило модификации весов:

$$\textbf{если } \langle w, x_i \rangle y_i < 0 \textbf{ то } w := w + \eta x_i y_i, \tag{1.3}$$

называемое *правилом Хэбба*. Соответственно, в Алгоритме 1.1 заменяется шаг 4.

Для правила Хэбба докажем теорему сходимости. Она справедлива не только для бинарных, но и для произвольных действительных признаков.

Теорема 1.1 (Новиков, 1962 [14]). *Пусть* $X = \mathbb{R}^{n+1}$, $Y = \{-1, 1\}$, *и выборка* X^ℓ *линейно разделима — существует вектор* $\tilde{w}$ *и положительное число* δ *такие, что* $\langle \tilde{w}, x_i \rangle y_i > \delta$ *для всех* $i = 1, \ldots, \ell$. *Тогда Алгоритм 1.1 сходится за конечное число шагов к вектору весов, разделяющему обучающие объекты без ошибок, из любого начального положения* w^0, *при любом положительном* η, *независимо от порядка предъявления объектов. Если* $w^0 = 0$, *то достаточное число исправлений вектора весов не превосходит*

$$t_{\max} = \left(\frac{D}{\delta}\right)^2, \quad \text{где } D = \max_{x \in X^\ell} \|x\|.$$

Метод стохастического градиента. Исходя из принципа минимизации эмпирического риска задача настройки синаптических весов может быть сведена к поиску вектора w, доставляющего минимум функционалу качества:

$$Q(w) = \sum_{i=1}^{\ell} \mathscr{L}\big(a(x_i), y_i\big) \to \min_w, \tag{1.4}$$

где $\mathscr{L}(a, y)$ — заданная функция потерь, характеризующая величину ошибки ответа a при правильном ответе y. Применим для минимизации $Q(w)$ метод градиентного спуска. Запишем правило изменения вектора весов на каждой итерации:

$$w := w - \eta \frac{\partial Q}{\partial w},$$

где $\eta > 0$ — величина шага в направлении антиградиента. Предполагая, что функция потерь $\mathscr{L}$ и функция активации φ дифференцируемы, распишем градиент:

$$w := w - \eta \sum_{i=1}^{\ell} \mathscr{L}'_a\big(a(x_i), y_i\big)\varphi'\big(\langle w, x_i\rangle\big)x_i.$$

Алгоритм 1.2. Обучение персептрона методом стохастического градиента.

Вход:

X^ℓ — обучающая выборка;

η — темп обучения;

Выход:

Синаптические веса $w_0, w_1, \ldots, w_n$;

1: инициализировать веса:
$w_j := \text{random}\left(-\frac{1}{2n}, \frac{1}{2n}\right)$;

2: инициализировать текущую оценку функционала:
$Q := \sum_{i=1}^{\ell} \mathscr{L}\big(a(x_i), y_i\big)$;

3: **повторять**

4: выбрать объект x_i из X^ℓ случайным образом;

5: вычислить выходное значение алгоритма $a(x_i)$ и ошибку:
$\varepsilon_i := \mathscr{L}\big(a(x_i), y_i\big)$;

6: сделать шаг градиентного спуска:
$w := w - \eta \mathscr{L}'_a\big(a(x_i), y_i\big)\varphi'(\langle w, x_i\rangle)x_i$;

7: оценить значение функционала:
$Q := \frac{\ell-1}{\ell} Q + \frac{1}{\ell}\varepsilon_i^2$;

8: **пока** значение Q не стабилизируется;

Вычислительные возможности нейронных сетей. Возникает вопрос: любую ли функцию можно представить (хотя бы приближённо) с помощью нейронной сети? Следующие факты позволяют ответить на этот вопрос утвердительно.

1. Любая булева функция представима в виде двухслойной сети. Это тривиальное следствие нейронной представимости функций И, ИЛИ, НЕ и представимости произвольной булевой функции в виде дизъюнктивной нормальной формы [5].

2. Из простых геометрических соображений вытекает, что двухслойная сеть с пороговыми функциями активации позволяет выделить произвольный выпуклый многогранник в n-мерном пространстве признаков. Трёхслойная сеть позволяет вычислить любую конечную линейную комбинацию характеристических функций выпуклых многогранников, следовательно, аппроксимировать любые области с непрерывной границей, включая неодносвязные, а также аппроксимировать любые непрерывные функции.

Теорема 1.2 (Колмогоров, 1957). *Любая непрерывная функция n аргументов на единичном кубе $[0,1]^n$ представима в виде суперпозиции непрерывных функций одного аргумента и операции сложения:*

$$f(x^1, x^2, \ldots, x^n) = \sum_{k=1}^{2n+1} h_k\left(\sum_{i=1}^{n} \varphi_{ik}(x^i)\right),$$

где h_k, φ_{ik} — непрерывные функции, причём φ_{ik} не зависят от выбора f.

Алгоритм 1.3. Обучение двухслойной сети методом back-propagation — обратного распространения ошибки

Вход:

$X^\ell = (x_i, y_i)_{i=1}^\ell$ — обучающая выборка, $x_i \in \mathbb{R}^n$, $y_i \in \mathbb{R}^M$;

H — число нейронов в скрытом слое;

η — темп обучения;

Выход:

синаптические веса w_{jh}, w_{hm};

1: инициализировать веса небольшими случайными значениями:
$w_{jh} := \text{random}\left(-\frac{1}{2n}, \frac{1}{2n}\right)$;
$w_{hm} := \text{random}\left(-\frac{1}{2H}, \frac{1}{2H}\right)$;

2: **повторять**

3: выбрать объект x_i случайным образом;

4: прямой ход:
$u_i^h := \sigma_h\left(\sum_{j=0}^J w_{jh} v^j(x_i)\right)$, для всех $h = 1, \ldots, H$;
$a_i^m := \sigma_m\left(\sum_{h=0}^H w_{hm} u^h(x_i)\right)$, для всех $m = 1, \ldots, M$;
$\varepsilon_i^m := a_i^m - y_i^m$, для всех $m = 1, \ldots, M$;
$Q_i := \sum_{m=1}^M (\varepsilon_i^m)^2$;

5: обратный ход:
$\varepsilon_i^h := \sum_{m=1}^M \varepsilon_i^m \sigma'_m w_{hm}$, для всех $h = 1, \ldots, H$;

6: градиентный шаг:
$w_{hm} := w_{hm} - \eta \varepsilon_i^m \sigma'_m u^h$, для всех $h = 0, \ldots, H$, $m = 1, \ldots, M$;
$w_{jh} := w_{jh} - \eta \varepsilon_i^h \sigma'_h x^j$, для всех $j = 0, \ldots, n$, $h = 1, \ldots, H$;

7: $Q := \frac{\ell-1}{\ell} Q + \frac{1}{\ell} Q_i$;

8: **пока** Q не стабилизируется;

Достоинства метода обратного распространения.

- Достаточно высокая эффективность. Прямой ход, обратный ход и вычисления градиента требуют порядка $O(Hn + HM)$ операций.
- Через каждый нейрон проходит информация только о связных с ним нейронах. Поэтому back-propagation легко реализуется на вычислительных устройствах с параллельной архитектурой.
- Высокая степень общности. Алгоритм легко записать для произвольного числа слоёв, произвольной размерности выходов и выходов, произвольной функции потерь и произвольных функций активации, возможно, различных у разных нейронов. Кроме того, back-propagation не накладывает никаких ограничений на используемый метод оптимизации. Его можно применять вместе с методом скорейшего спуска, сопряженных градиентов, Ньютона-Рафсона и др.

Недостатки метода обратного распространения.

- Метод не всегда сходится. Для улучшения сходимости приходится применять большое количество различных эвристических ухищрений.
- Процесс градиентного спуска склонен застревать в многочисленных локальных минимумах функционала Q.
- Приходится заранее фиксировать число нейронов скрытого слоя H. В то же время, это критичный параметр сложности сети, от которого может существенно зависеть качество обучения и скорость сходимости.
- При чрезмерном увеличении числа весов сеть склонна к переобучению.
- Если применяются функции активации с горизонтальными асимптотами, типа сигмоидной или th, то сеть может попадать в состояние «паралича». Чем больше значения синаптических весов на входе нейрона, тем ближе значение производной σ' к нулю, тем меньше изменение синаптических весов в соответствии с формулами (1.8)–(1.9). Если нейрон один раз попадает в такую «мёртвую зону», то у него практически не остаётся шансов из неё выбраться. Парализоваться могут отдельные связи, нейроны, или вся сеть в целом.