Нейронные сети. Общая структура (особый класс функций для оптимизации). Back propagation как вычислительный подход.

Волканова Маргарита, Третьякова Александра, Фёдоров Никита

Санкт-Петербургский государственный университет
Математико-механический факультет
Кафедра статистического моделирования

СтатМод

Санкт-Петербург
2019г.

Постановка задачи аппроксимации особым классом функций

- X — множество объектов, Y — множество ответов;
- $X^n = (x_i, y_i)_{i=1}^n$ — обучающая выборка, $x_i \in \mathbb{R}^p$;
- $(x^1, \dots, x^p)$ — признаки объекта $x \in X$;

Рассмотрим стандартное построение предсказывающей модели:

$$Q(a, X^n) = \frac{1}{n} \sum_{i=1}^{n} L(a, x_i, y_i) \to \min_{w},$$

где $a(x, w) = \sigma(\langle w, x \rangle) = \sigma \left(\sum_{j=1}^{p} w_j x^j - w_0 \right),$

$w_k \in \mathbb{R}$, $k = 0, \dots, p$; $\sigma : \mathbb{R} \to \mathbb{R}$ — функция активации;
$L(a, x_i, y_i)$ — функция потерь, $i = 1, \dots, n$.

Задача классификации: $Y = \{\pm 1\}$, $a(x_j, w) = sign\langle w, x_j \rangle$,

$$Q(w, X^n) = \sum_{j=1}^{n} L\left(a(x_j, w), y_j\right) = \sum_{j=1}^{n} [y_j \langle w, x_j \rangle < 0] \to \min_{w}.$$

Задача регрессии: $Y = \mathbb{R}$, $a(x_j, w) = \sigma(\langle w, x_j \rangle)$,

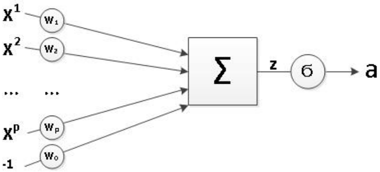
$$Q(w, X^n) = \sum_{j=1}^{n} L\left(a(x_j, w), y_j\right) = \sum_{j=1}^{n} \left(\sigma(\langle w, x_j \rangle) - y_j\right)^2 \to \min_{w}.$$

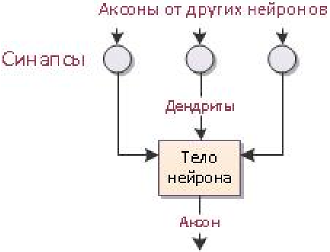
При $\sigma(z) = z$ получаем многомерную линейную регрессию.

Модель нейрона МакКаллока–Питтса

$$a(x, w) = \sigma(\langle w, x \rangle) = \sigma\left(\sum_{j=1}^{p} w_j x^j - w_0\right),$$

- $x^j \in \mathbb{R}^n$, $j = 1, \ldots, p$, — числовые признаки, входы;
- $w_j \in \mathbb{R}$, $j = 1, \ldots, p$ — весовые коэффициенты (синаптические веса);
- $\sigma(z)$ — функция активации;
- w_0 — порог активации (смещение).



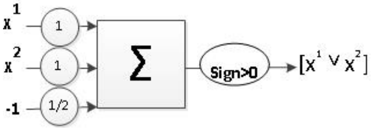


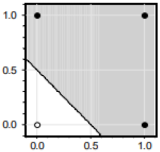
Булевы операции в виде нейронов

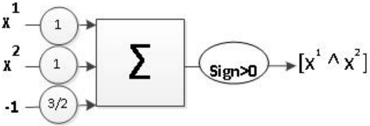
Унарная булева операция: $\neg x^1 = \left[-x^1 + \frac{1}{2} > 0\right]$

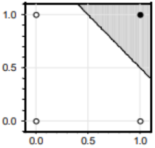
Бинарные булевы операции:

$$x^1 \vee x^2 = \left[x^1 + x^2 - \frac{1}{2} > 0\right] \qquad x^1 \wedge x^2 = \left[x^1 + x^2 - \frac{3}{2} > 0\right]$$



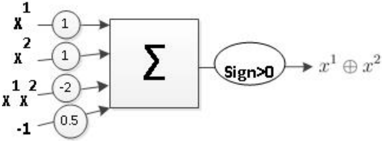


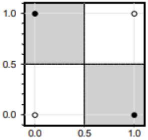




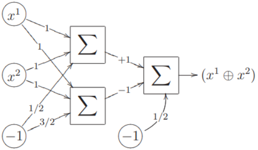
Функцию XOR не реализовать одним нейроном с двумя входами:

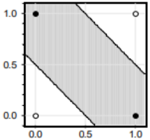
1. Добавление нелинейного признака:
$x^1 \oplus x^2 = \left[x^1 + x^2 - 2x^1x^2 - \frac{1}{2} > 0\right]$





2. Сеть (двухслойная суперпозиция) функций И, ИЛИ, НЕ:
$x^1 \oplus x^2 = \left[\neg\left(x^1 \wedge x^2 - x^1 \vee x^2\right) > 0\right].$





Нейронная сеть — суперпозиция нейронов.

Любую ли функцию можно представить нейросетью?

Теорема (Цыбенко, 1989)

Пусть $\sigma(x)$ *— непостоянная, ограниченная и монотонно возрастающая непрерывная функция;* $C(I_{p_0})$ *— множество непрерывных функций на* $[0,1]^{p_0}$.
Тогда для любой $f \in C(I_{p_0})$ *и* $\varepsilon > 0$ *существуют* $p_1 \in \mathbb{Z}$ *и* α_i, b_i, $w_{ij} \in \mathbb{R}$, $i = 1,\ldots,p_1$, $j = 1,\ldots,p_0$, *такие что для любого* $x = (x^1,\ldots,x^{p_0}) \in I_{p_0}$ *выполняется*

$$|F(x^1,\ldots,x^{p_0}) - f(x^1,\ldots,x^{p_0})| < \varepsilon,$$

$$\text{где } F(x^1,\ldots,x^{p_0}) = \sum_{i=1}^{p_1} \alpha_i \, \sigma\left(\sum_{j=1}^{p_0} w_{ij}x^j + b_i.\right)$$

Вывод: С помощью линейных операций и одной нелинейной функции активации можно приблизить любую непрерывную функцию с любой желаемой точностью.

Функции активации:

- Сигмоида: $\sigma(z) = \frac{1}{1+e^{-a\,z}}$, $a \in \mathbb{R}$;
- Softmax: $SM_i(z) = \frac{e^{z_i}}{\sum_{k=1}^{K} e^{z_k}}$;
- Гиперболический тангенс: $\sigma(z) = \frac{e^{a\,z} - e^{-a\,z}}{e^{a\,z} + e^{-a\,z}}$, $a \in \mathbb{R}$;
- Выпрямитель: $ReLU(p) = \max(0, p)$;

Функции потерь:(t - истинное значение, $p = \sigma(y)$)

- Среднеквадратичная ошибка: $MSE(p,t) = (p-t)^2$;
- Бинарная кросс-энтропия: $BCE(p,t) = -t \log p - (1-t)\log(1-p)$;
- Кросс энтропия: $CE(p,t) = -\sum_{c=1}^{N} t_c \log p_c$;

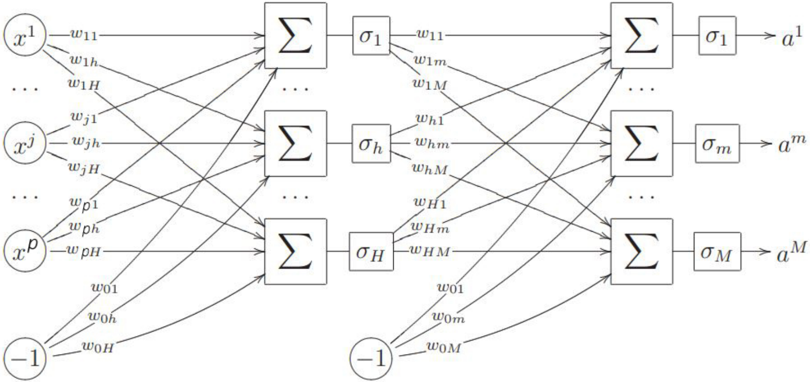
Многослойная нейронная сеть

Пусть для общности $Y = \mathbb{R}^M$ и слоёв для простоты только два.

входной слой, p признаков

скрытый слой, H нейронов

выходной слой, M нейронов



Напоминание: Стохастический градиентный спуск

Идея: на каждом шаге учитывать только одно наблюдение.

$$Q(w) = \sum_{j=1}^{l} L\left(w, x_j, y_j\right) \to \min_w .$$

Вход: $X^n = (x_i, y_i)_{i=1}^n$, λ, η;

Выход: синаптические веса

$w \equiv (w_{jh}, w_{hm}) = (\{w_{jh}\}_{j,h=0}^{p,H}, \{w_{hm}\}_{h,m=0}^{H,M}) \in \mathbb{R}^{H(p+M+1)+M}$;

1. Инициализация веса w, выбор скорости обучения η, темп забывания λ, начальная оценка функционала $\overline{Q}(w) = \frac{1}{n} Q(w) = \frac{1}{n}\sum_{i=1}^n L(w, x_i, y_i)$;
2. **повторять**
    1. Случайный выбор x_i из X^n и вычисление функции потерь $L_i := L(w, x_i, y_i)$;
    2. Градиентный шаг: $w := w - \eta \nabla L(w, x_i, y_i)$;
    3. Оценка функционала $Q := (1-\lambda)Q + \lambda L_i$;

пока значение Q и/или w не стабилизируется.

Дифференцирование суперпозиции функций

Выходные значения сети $a^m(x_i)$, $m = 1, \ldots, M$ на объекте x_i:

$$a^m(x_i) = \sigma_m\left(\sum_{h=0}^{H} w_{hm} u^h(x_i)\right); \qquad u^h(x_i) = \sigma_h\left(\sum_{j=0}^{p} w_{jh} x_i^j\right).$$

Пусть для конкретности $L_i(w)$ — средний квадрат ошибки:

$$L_i(w) = \frac{1}{2}\sum_{m=1}^{M}\left(a^m(x_i) - y_i^m\right)^2.$$

Промежуточная задача: найти частые производные

$$\frac{\partial L_i(w)}{\partial a^m}; \quad \frac{\partial L_i(w)}{\partial u^h}.$$

Промежуточная задача: частая производная

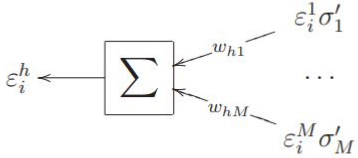
$$\frac{\partial L_i(w)}{\partial a^m} = a^m(x_i) - y_i^m = \varepsilon_i^m$$

— это ошибка на выходном слое.

$$\frac{\partial L_i(w)}{\partial u^h} = \sum_{m=1}^{M} (a^m(x_i) - y_i^m)\sigma_m^{'} w_{hm} = \sum_{m=1}^{M} \varepsilon_i^m \sigma_m^{'} w_{hm} = \varepsilon_i^h$$

— назовём это *ошибкой на скрытом слое*.

Похоже, что ε_i^h вычисляется по ε_i^m, если запустить сеть «задом наперёд»:



Быстрое вычисление градиента

Теперь, имея частные производные $L_i(w)$ по a^m и u^h, легко выписать градиент $L_i(w)$ по весам w:

$$\frac{\partial L_i(w)}{\partial w_{hm}} = \frac{\partial L_i(w)}{\partial a^m} \frac{\partial a^m}{\partial w_{hm}} = \varepsilon_i^m \sigma_m^{'} u^h(x_i), \quad h = 0, \ldots, H, \quad m = 1, \ldots, M;$$

$$\frac{\partial L_i(w)}{\partial w_{jh}} = \frac{\partial L_i(w)}{\partial u^h} \frac{\partial u^h}{\partial w_{jh}} = \varepsilon_i^h \sigma_h^{'} x_i^j, \quad j = 0, \ldots, p, \quad h = 1, \ldots, H.$$

Алгоритм обратного распространения ошибки BackProp

Вход: $X^n = (x_i, y_i)_{i=1}^n \subset \mathbb{R}^p \times \mathbb{R}^M$; параметры H, λ, η.
Выход: синаптические веса w_{jh}, w_{hm}.

1. инициализировать веса w_{jh}, w_{hm};
2. **повторять**
   1. случайно выбрать элемент x_i из выборки X^n;
   2. прямой ход:
      $u_i^h := \sigma_h \left(\sum_{j=0}^{p} w_{jh} x_i^j \right), \; h = 1, \dots, H;$
      $a_i^m := \sigma_m \left(\sum_{h=0}^{H} w_{hm} u_i^h \right), \; \varepsilon_i^m := a_i^m - y_{im}, \; m = 1, \dots, M;$
      $L_i := \sum_{m=1}^{M} (\varepsilon_i^m)^2$, вычисление производных $\sigma_m^{'}, \sigma_h^{'}$;
   3. обратный ход:
      $\varepsilon_i^h := \sum_{m=1}^{M} \varepsilon_i^m \sigma^{'} w_{hm}, \; h = 1, \dots, H;$
   4. градиентный шаг:
      $w_{hm} := w_{hm} - \eta \varepsilon_i^m \sigma_m^{'} u_i^h, \; h = 0, \dots, H, \; m = 1, \dots, M;$
      $w_{jh} := w_{jh} - \eta \varepsilon_i^h \sigma_h^{'} x_i^j, \; j = 0, \dots, p, \; h = 1, \dots, H;$
   5. $Q := (1 - \lambda) Q + \lambda L_i$;

пока Q не стабилизируется.

Преимущества и недостатки

Преимущества:

- эффективность: быстрое вычисление градиента;
- метод легко обобщается на любые σ, L;
- возможно динамическое (потокое) обучение;
- на сверхбольших выборках не обязательно брать все x_i;
- возможность распараллеливания.

Недостатки (есть все те же, что и у SG):

- метод не всегда сходится;
- возможна медленная сходимость;
- застревание в локальных минимумах;
- проблема переобучения;
- сложно подбирать эвристики.

Улучшение сходимости и качества градиентного обучения

- **Эвристики:** (те же, что и в обычном SG)
  - инициализация весов;
  - порядок предъявления объектов;
  - оптимизация величины градиентного шага;
  - регуляризация (сокращение весов).
- **Более тщательный подбор начального приближения:** Нейроны первого слоя настраиваются как H отдельных однослойных сетей:
  - либо по случайной подвыборке $X^{'} \subseteq X^n$;
  - либо по случайному подмножеству входов;
  - либо из различных случайных начальных приближений;

  тем самым обеспечивается *различность нейронов*.
- Выбивание из локальных минимумов (jogging of weights).

1. **Адаптивный градиентный шаг**. На каждом шаге ищем η_*:

$$Q(w - \eta \frac{\delta Q}{\delta w}) \to \min_{\eta}.$$

2. **Диагональный метод Левенберга-Марквардта.**
Метод Ньютона-Рафсона (второго порядка):

$$w := w - \eta (Q''(w))^{-1} Q'(w),$$

где $Q''(w) = \left(\frac{\partial^2 Q(w)}{\partial w_{jh} \partial w_{j'h'}} \right)$ — гессиан размера $(H(p + M + 1) + M)^2$.
Эвристика. Считаем, что гессиан диагонален:

$$w_{jh} := w_{jh} - \eta \left(\frac{\partial^2 Q(w)}{\partial w_{jh}^2} + \mu \right)^{-1} \frac{\partial Q(w)}{\partial w_{jh}},$$

η — темп обучения,
μ — параметр, предотвращающий обнуление элемента.

Оптимизация структуры сети

1. **Выбор числа слоёв.**
Если знаем, что классы линейно разделимы, то можно ограничиться одним слоем. Двух-трёх слоёв обычно достаточно.

2. **Выбор числа нейронов в скрытом слое H.**

1. **Визуальный способ.** Если граница классов (или кривая регрессии) слишком сглажена — количество нейронов в слое нужно увеличить, а если есть резкие колебания, то, наоборот, уменьшить (для задач с небольшим числом признаков).
2. **По внешнему критерию.**
   - Средняя ошибка на тестовой выборке;
   - Cross-validation;

   Недостаток этого способа — высокая трудоёмкость.

Динамическое наращивание сети

1. Обучение сети при заведомо недостаточном числе нейронов $H \ll n$, пока ошибка не перестаёт убывать;
2. Добавление нового нейрона и его инициализация путем обучения
   - либо по случайной подвыборке $X' \subseteq X^n$;
   - либо по объектам с наибольшими значениями потерь;
   - либо по случайному подмножеству входов;
   - либо из различных случайных начальных приближений.
3. Снова итерации BackProp;

Эмпирический опыт:

- После добавления новых нейронов ошибка, обычно, сначала резко возрастает, затем быстро сходится к меньшему значению.
- Общее время обучения обычно лишь в 1.5–2 раза больше, чем если бы в сети сразу было нужное количество нейронов. Полезная информация, накопленная сетью, не теряется при добавлении новых нейронов.
- Полезно наблюдать за внешним критерием: прохождение $Q(X^k)$ через минимум является надежным критерием останова.

Удаление избыточных связей (OBD — Optimal Brain Damage)

Пусть w — локальный минимум $Q(w)$,
тогда $Q(w)$ можно аппроксимировать квадратичной формой:

$$Q(w+\delta) = Q(w) + \frac{1}{2}\delta^{\mathrm{T}} Q''(w)\delta + o(\|\delta\|^2),$$

где $Q''(w) = \frac{\partial^2 Q(w)}{\partial w_{jh} \partial w_{j'h'}}$ — гессиан размера $(H(p+M+1)+M)^2$.

Эвристика. Пусть гессиан $Q''(w)$ диагонален, тогда

$$\delta^{\mathrm{T}} Q''(w)\delta = \sum_{j=0}^{p}\sum_{h=0}^{H} \delta_{jh}^2 \frac{\partial^2 Q(w)}{\partial w_{jh}^2} + \sum_{h=0}^{H}\sum_{m=0}^{M} \delta_{hm}^2 \frac{\partial^2 Q(w)}{\partial w_{hm}^2}.$$

Хотим обнулить вес: $w_{jh} + \delta_{jh} = 0$. Как изменится $Q(w)$?

Определение

Значимость (salience) веса w_{jh} *— это изменение функционала* $Q(w)$ *при его обнулении:* $S_{jh} = w_{jh}^2 \frac{\partial^2 Q(w)}{\partial w_{jh}^2}$.

Удаление избыточных связей (OBD — Optimal Brain Damage)

1. В BackProp вычислять вторые производные $\frac{\partial^2 Q(w)}{\partial w_{jh}^2}$, $\frac{\partial^2 Q(w)}{\partial w_{hm}^2}$.
2. Если процесс минимизации $Q(w)$ пришел в минимум, то
   - упорядочить веса по убыванию S_{jh};
   - удалить d связей с наименьшей значимостью;
   - снова запустить BackProp.
3. Если $Q(w, X^n)$ или $Q(w, X^k)$ существенно ухудшился, то вернуть последние удаленные связи и выйти.

Аналогично, OBD можно использовать для отбора информативных признаков для нейрона скрытого слоя. Суммарная значимость признака: $S_j = \sum_{h=1}^{H} S_{jh}$.

Эмпирический опыт: сеть, построенная с помощью OBD, меньше склонна к переобучению, чем сеть сразу построенная по полученной структуре (со случайно инициализированными весами).