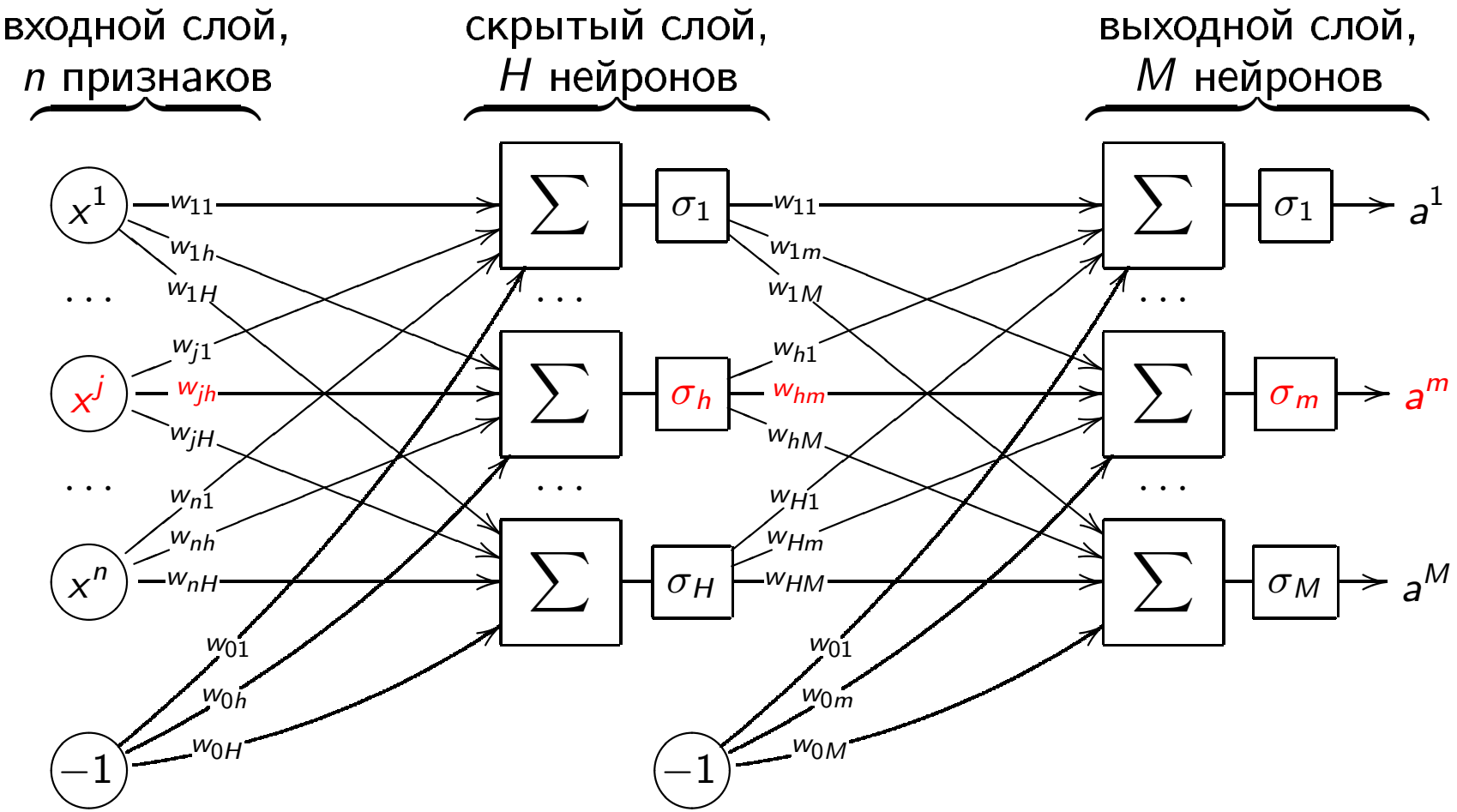
Пусть для общности $Y = \mathbb{R}^M$, для простоты слоёв только два:

$$a^m(x, w) = \sigma_m\left(\sum_{h=0}^{H} w_{hm}\sigma_h\left(\sum_{j=0}^{n} w_{jh}f_j(x_i)\right)\right).$$



Напоминание: Алгоритм SG (Stochastic Gradient)

Задача минимизации суммарных потерь:

$$Q(w) := \sum_{i=1}^{\ell} \mathscr{L}_i(w) \to \min_w .$$

Вход: выборка X^ℓ; темп обучения η; параметр λ;
Выход: веса $w \equiv \left(w_{jh}, w_{hm}\right) \in \mathbb{R}^{H(n+M+1)+M}$;

1: инициализировать веса w и текущую оценку $Q(w)$;
2: **повторять**
3: выбрать объект x_i из X^ℓ (например, случайно);
4: вычислить потерю $\mathscr{L}_i := \mathscr{L}_i(w)$;
5: градиентный шаг: $w := w - \eta \nabla \mathscr{L}_i(w)$;
6: оценить значение функционала: $Q := (1-\lambda)Q + \lambda \mathscr{L}_i$;
7: **пока** значение Q и/или веса w не стабилизируются;

Задача дифференцирования суперпозиции функций

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

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

Пусть для конкретности $\mathscr{L}_i(w)$ — средний квадрат ошибки:

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

Промежуточная задача: найти частные производные

$$\frac{\partial \mathscr{L}_i(w)}{\partial a^m}; \qquad \frac{\partial \mathscr{L}_i(w)}{\partial u^h}.$$

Промежуточная задача: частные производные

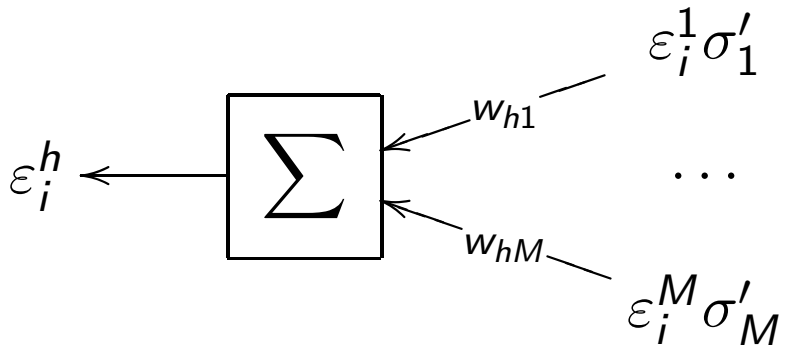
$$\frac{\partial \mathscr{L}_i(w)}{\partial a^m} = a^m(x_i) - y_i^m = \varepsilon_i^m$$

— это ошибка на выходном слое;

$$\frac{\partial \mathscr{L}_i(w)}{\partial u^h} = \sum_{m=1}^{M} \big(a^m(x_i) - y_i^m\big)\sigma'_m w_{hm} = \sum_{m=1}^{M} \varepsilon_i^m \sigma'_m w_{hm} = \varepsilon_i^h$$

— назовём это *ошибкой на скрытом слое*.

Похоже, что ε_i^h вычисляется по ε_i^m путём его пропускания через сеть в обратном направлении:



Быстрое вычисление градиента

Теперь, имея частные производные $\mathscr{L}_i(w)$ по a^m и u^h, легко выписать градиент $\mathscr{L}_i(w)$ по весам w:

$$\frac{\partial \mathscr{L}_i(w)}{\partial w_{hm}} = \frac{\partial \mathscr{L}_i(w)}{\partial a^m}\frac{\partial a^m}{\partial w_{hm}} = \varepsilon_i^m \sigma'_m u^h(x_i), \quad m = 1..M, \quad h = 0..H;$$

$$\frac{\partial \mathscr{L}_i(w)}{\partial w_{jh}} = \frac{\partial \mathscr{L}_i(w)}{\partial u^h}\frac{\partial u^h}{\partial w_{jh}} = \varepsilon_i^h \sigma'_h f_j(x_i), \quad h = 1..H, \quad j = 0..n;$$

Алгоритм обратного распространения ошибки BackProp:

Вход: $X^\ell = (x_i, y_i)_{i=1}^\ell \subset \mathbb{R}^n \times \mathbb{R}^M$; параметры H, λ, η;

Выход: синаптические веса w_{jh}, w_{hm};

1: ...

Алгоритм BackProp

1: инициализировать веса w_{jh}, w_{hm};
2: **повторять**
3: выбрать объект x_i из X^ℓ (например, случайно);
4: прямой ход:
$u_i^h := \sigma_h\big(\sum_{j=0}^{J} w_{jh} x_i^j\big)$, $h = 1..H$;
$a_i^m := \sigma_m\big(\sum_{h=0}^{H} w_{hm} u_i^h\big)$, $\varepsilon_i^m := a_i^m - y_i^m$, $m = 1..M$;
$\mathscr{L}_i := \sum_{m=1}^{M} (\varepsilon_i^m)^2$;
5: обратный ход: $\varepsilon_i^h := \sum_{m=1}^{M} \varepsilon_i^m \sigma'_m w_{hm}$, $h = 1..H$;
6: градиентный шаг:
$w_{hm} := w_{hm} - \eta \varepsilon_i^m \sigma'_m u_i^h$, $h = 0..H$, $m = 1..M$;
$w_{jh} := w_{jh} - \eta \varepsilon_i^h \sigma'_h x_i^j$, $j = 0..n$, $h = 1..H$;
7: $Q := (1 - \lambda) Q + \lambda \mathscr{L}_i$;
8: **пока** Q не стабилизируется;

Алгоритм BackProp: преимущества и недостатки

Преимущества:

- быстрое вычисление градиента;
- метод легко обобщается на любые σ, $\mathscr{L}$;
- возможно динамическое (потоковое) обучение;
- на сверхбольших выборках не обязательно брать все x_i;
- возможность распараллеливания;

Недостатки — все те же, свойственные SG:

- возможна медленная сходимость;
- застревание в локальных минимумах;
- проблема «паралича сети» (горизонтальные асимптоты σ);
- проблема переобучения;
- подбор комплекса эвристик является искусством;