

Основы практического использования нейронных сетей.

Лекция 1. Функции активации. Функция ошибки.

Дмитрий Буряк.
к.ф.-м.н.
dyb04@yandex.ru

Основные темы спецкурса

- ❑ Построение и обучение НС
 - Оптимизация гиперпараметров НС
 - Выбор функции ошибки
 - Анализ, оптимизация обученной сети
- ❑ Особенности построения глубоких НС
 - Алгоритмы обучения
 - Проблемы при обучении
 - Методы регуляризации
- ❑ Обзор современных архитектур глубоких нейронных сетей.
- ❑ Программные средства проектирования и реализации НС.
- ❑ Практическое задание

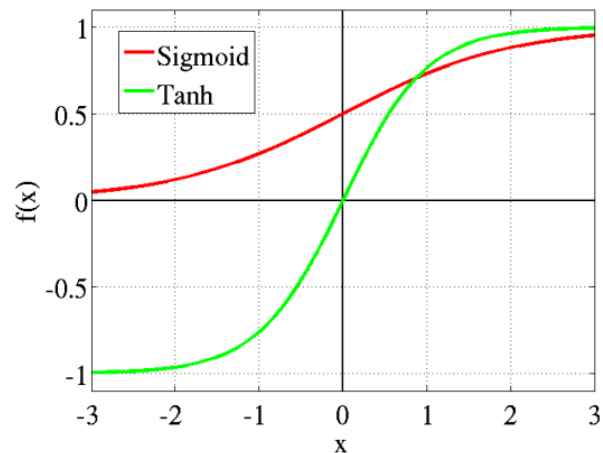
Литература

- ❑ <http://www.deeplearningbook.org/>
- ❑ <http://cs231n.github.io/>
- ❑ <http://neuralnetworksanddeeplearning.com/index.html>
- ❑ Франсуа Шолле. Глубокое обучение на Python, 2018.
Francois Chollet. Deep Learning with Python, 2nd edition, 2021
- ❑ Eli Stevens, et al., Deep Learning with PyTorch, 2020.

Функция активации.

Сигмоида

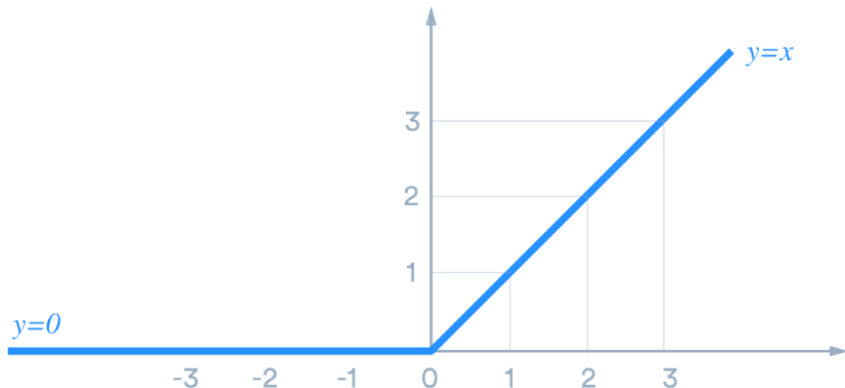
- Логистическая сигмоида, гиперболический тангенс
 - большая область насыщения → падение эффективности градиентных методов обучения;
 - проблема «исчезающего градиента» (vanishing gradient)
 - предпочтение гиперболическому тангенсу
 - редкое применение на фоне ReLU для сетей прямого распространения во внутренних слоях
 - используются в выходных слоях
 - активное использование в рекуррентных сетях, вероятностных моделях и др.



ReLU

□ ReLU:

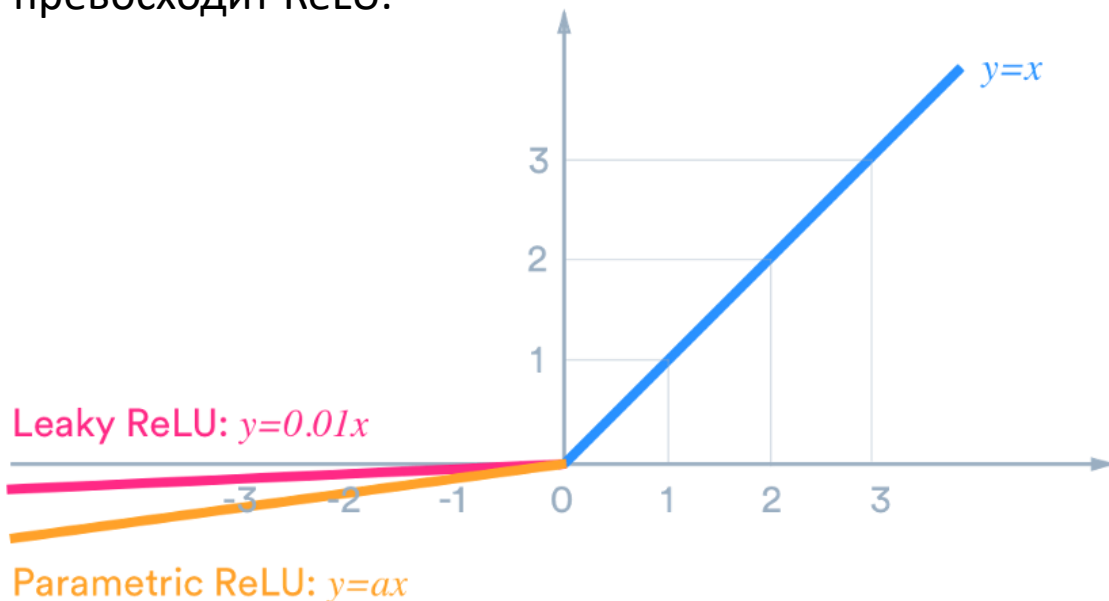
- $g(z) = \max\{0, z\}$ $z = \mathbf{W}^\top \mathbf{h} + b$.
- Низкая вычислительная сложность;
- Хорошая сходимость градиентных методов оптимизации;
- Постоянная производная при $x > 0$;
- Дополнительная устойчивость к переобучению (производная равна 0 при $x < 0$);
- Проблема «умирающий ReLU» (решение – уменьшение коэффициента обучения)



Варианты ReLU

□ Варианты ReLU: leaky ReLU, parameteric ReLU

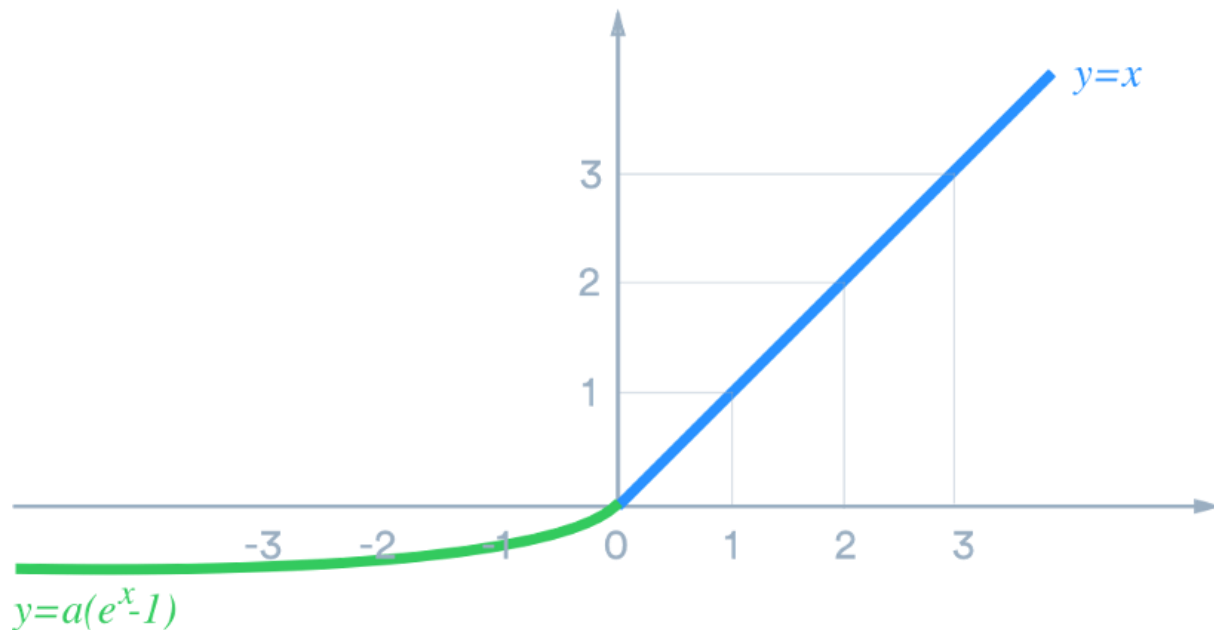
- $h_i = g(z, \alpha)_i = \max(0, z_i) + \alpha_i \min(0, z_i)$
- Решение проблемы «умирающий ReLU»;
- Не всегда превосходит ReLU.



ELU

□ Exponential Linear (ELU, SELU)

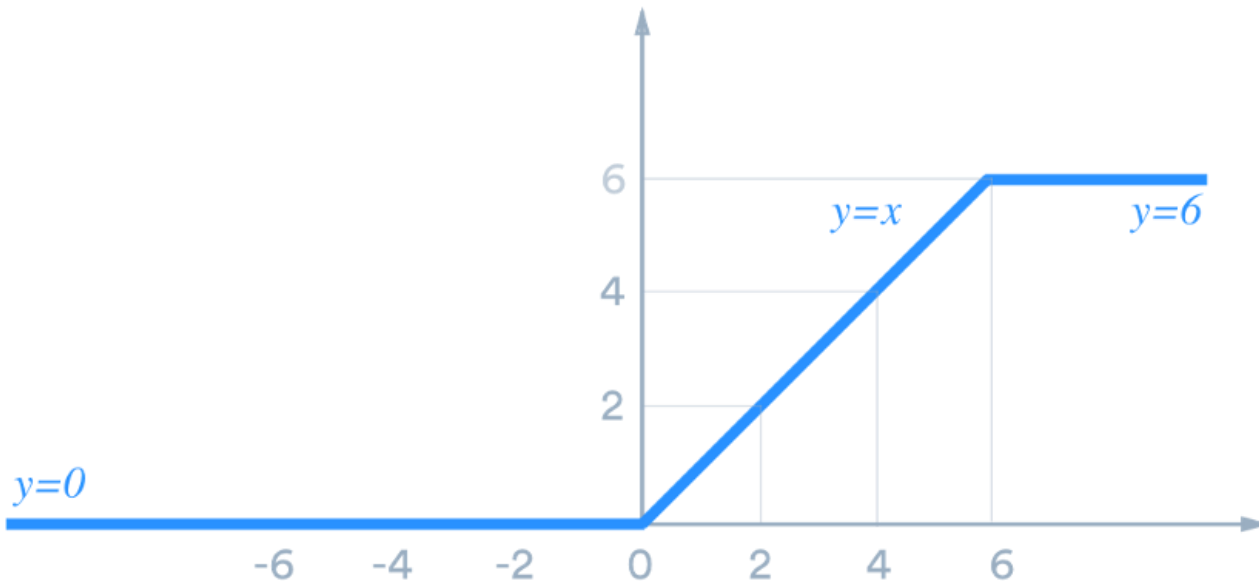
- Решение проблемы «умирающий ReLU»;
- Ограниченная снизу.



ReLU6

□ ReLU6

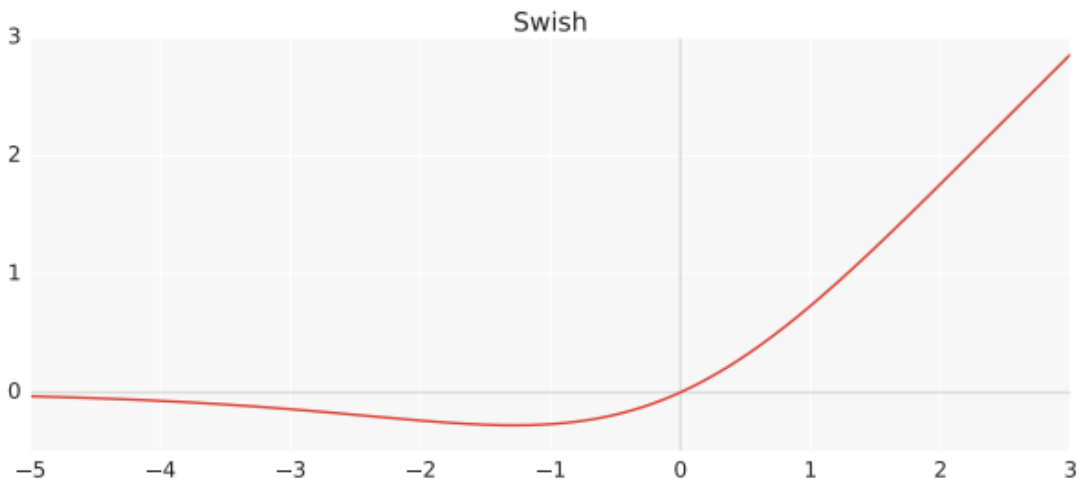
- Ограниченная сверху;
- Применяется в ряде известных архитектур (MobileNet)



Swish

□ Swish

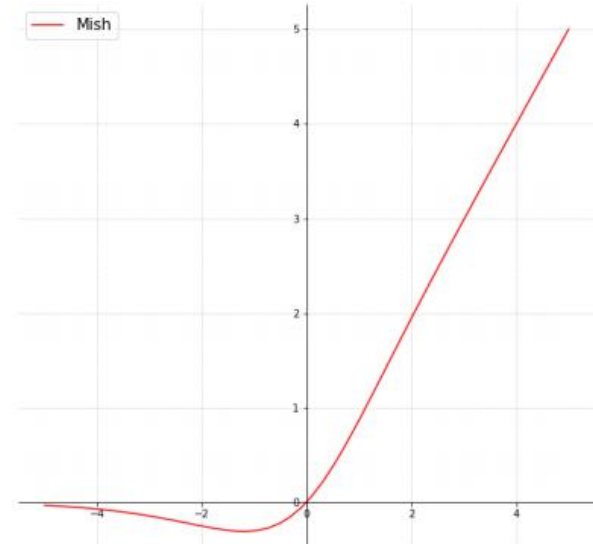
- $f(x) = x * \sigma(\beta x)$ β – фиксирован или обучаем
- Аналогична ReLU при $x > 0$;
- Решение проблемы «умирающей ReLU»;
- Немонотонная функция при $x < 0$;
- Во ряде случаев эффективнее ReLU.



Mish

□ Mish

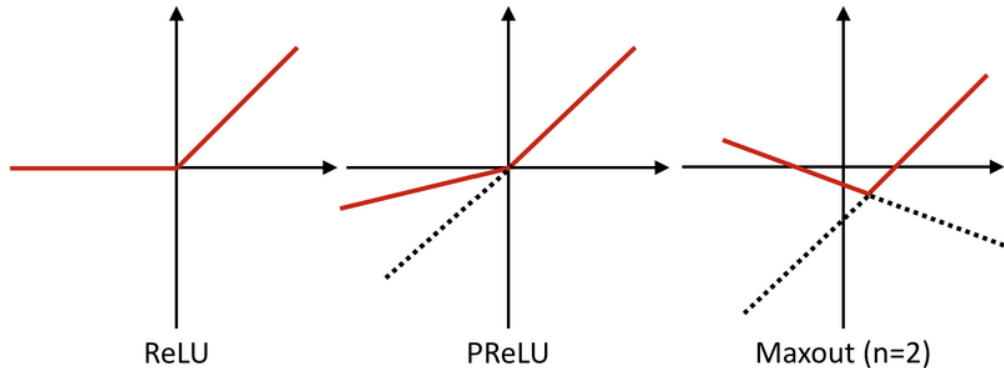
- $f(x) = x * \tanh(\text{softplus}(x))$ $\text{softplus}(x) = \ln(1 + e^x)$
- Аналогична ReLU при $x > 0$;
- Решение проблемы «умирающей ReLU»;
- Немонотонная функция при $x < 0$;
- Во ряде случаев эффективнее ReLU и Swish.



Maxout

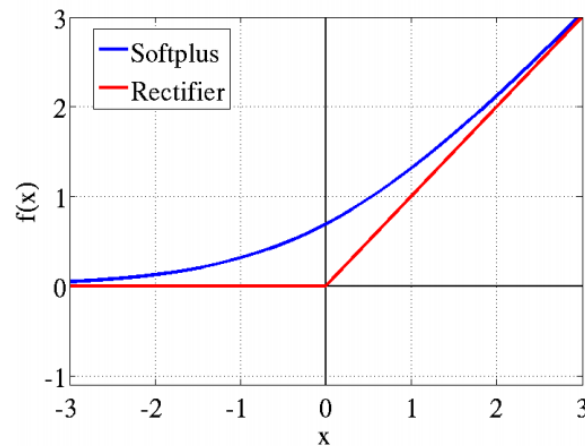
□ Maxout

- $\max(w_1^T x + b_1, w_2^T x + b_2)$
- Обобщение ReLU и Leaky ReLU
- Удваивает число параметров



Другие виды

- ❑ Другие виды функций активации:
 - многообразии функций имеют сходную эффективность
 - применение «неканонических» функций только при существенных отличиях от известных аналогов.
- ❑ Не использовать функцию активации
 - $h = g(W^T x + b) \rightarrow h = g(V^T U^T x + b)$
 - уменьшение числа весов: $np \rightarrow (n+p)q$
- ❑ Радиальные базисные функции (RBF)
 - $h_i = \exp\left(-\frac{1}{\sigma_i^2} \|W_{:,i} - x\|^2\right)$
 - Сложности при оптимизации.
- ❑ Softplus
 - $g(a) = \zeta(a) = \log(1 + e^a)$
 - Теоретически лучше ReLU, практически менее эффективен.



Функция ошибки.

Функция ошибки

$$E(X, Y, w) = C(X, Y, w) + \lambda R(w)$$

- ❑ C – функция потерь
 - X, Y – обучающее множество и желаемые выходы
 - w – настраиваемые параметры (веса, смещения)
 - $C \geq 0$
 - Вид функции потерь + функция активации в выходном слое → скорость сходимости алгоритма обучения.

- ❑ R – функция регуляризации

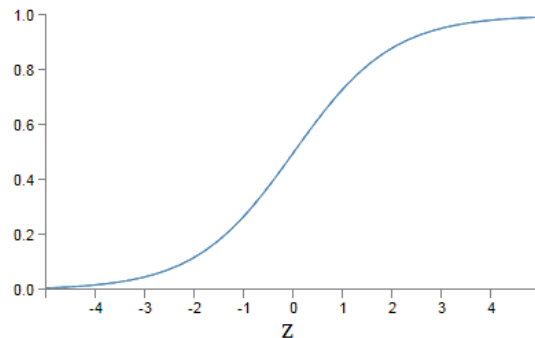
Среднеквадратичная функция (MSE)

$$C = \frac{1}{2n} \sum_i \|y_i - a_i\|^2$$

- ❑ 1 нейрон, 1 вход
- ❑ Функция активации - сигмоида

$$C = \frac{(y - a)^2}{2}$$

$$\frac{\partial C}{\partial w} = (a - y)\sigma'(z)x$$



- ❑ Зависимость от производной функции активации
- ❑ Большие ошибки ($a \approx 0, y = 1$; $a \approx 1, y = 0$) → область насыщения сигмоиды → замедление обучения
- ❑ MSE + сигмоида → низкая скорость сходимости алгоритма обучения

MSE + линейный нейрон

- ❑ Выходной слой – линейные нейроны

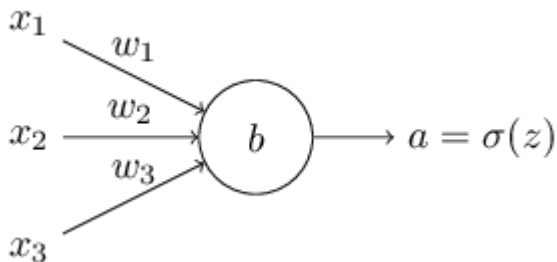
$$\frac{\partial C}{\partial w_{jk}^L} = \frac{1}{n} \sum_x a_k^{L-1} (a_j^L - y_j)$$

- ❑ Зависимость от ошибки на выходе
- ❑ MSE + линейная функция активации → высокая скорость сходимости алгоритма обучения

Cross Entropy

$$C = -\frac{1}{n} \sum_i \sum_j [y_{ij} \ln a_{ij} - (1 - y_{ij}) \ln(1 - a_{ij})]$$

□ Функция активации - сигмоида



$$C = -\frac{1}{n} \sum_x [y \ln a + (1 - y) \ln(1 - a)]$$

$$\frac{\partial C}{\partial w_j} = \frac{1}{n} \sum_x \frac{\sigma'(z) x_j}{\sigma(z)(1 - \sigma(z))} (\sigma(z) - y)$$

$$\sigma'(z) = \sigma(z)(1 - \sigma(z))$$

$$\frac{\partial C}{\partial w_j} = \frac{1}{n} \sum_x x_j (\sigma(z) - y)$$

□ Зависимость от ошибки на выходе

□ Cross entropy + сигмоида → высокая скорость сходимости алгоритма обучения

Функция правдоподобия

$$C \equiv -\ln a_y^L$$

- ❑ N выходов, функция активации – Softmax $a_j^L = \frac{e^{z_j^L}}{\sum_k e^{z_k^L}}$
- ❑ Входной вектор x относится к классу 1 $\rightarrow a_1^L \rightarrow 1 \rightarrow C \rightarrow \min$
или $\rightarrow a_1^L \rightarrow 0 \rightarrow C \rightarrow \max$

$$\frac{\partial C}{\partial w_{jk}^L} = a_k^{L-1} (a_j^L - y_j)$$

- ❑ Зависимость от ошибки на выходе
- ❑ Функция правдоподобия + softmax \rightarrow высокая скорость сходимости алгоритма обучения

Другие функции потерь

☐ Среднеквадратичная логарифмическая $\mathcal{L} = \frac{1}{n} \sum_{i=1}^n (\log(y^{(i)} + 1) - \log(\hat{y}^{(i)} + 1))^2$

☐ Средняя по модулю (MAE) $\mathcal{L} = \frac{1}{n} \sum_{i=1}^n |y^{(i)} - \hat{y}^{(i)}|$

☐ Средняя по модулю в процентах (MAPE) $\mathcal{L} = \frac{1}{n} \sum_{i=1}^n \left| \frac{y^{(i)} - \hat{y}^{(i)}}{y^{(i)}} \right| \cdot 100$

☐ Расстояние KL (Kullback Leibler) $\mathcal{L} = \underbrace{\frac{1}{n} \sum_{i=1}^n (y^{(i)} \cdot \log(y^{(i)}))}_{\text{entropy}} - \underbrace{\frac{1}{n} \sum_{i=1}^n (y^{(i)} \cdot \log(\hat{y}^{(i)}))}_{\text{cross-entropy}}$

☐ Hinge loss $\mathcal{L} = \frac{1}{n} \sum_{i=1}^n \max(0, 1 - y^{(i)} \cdot \hat{y}^{(i)})$

Выбор типа функции потерь

Задача регрессии

- функция активации в выходном слое: линейная
- функция потерь: MSE или Cross Entropy

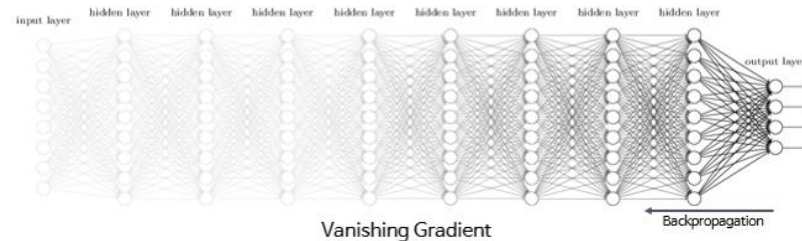
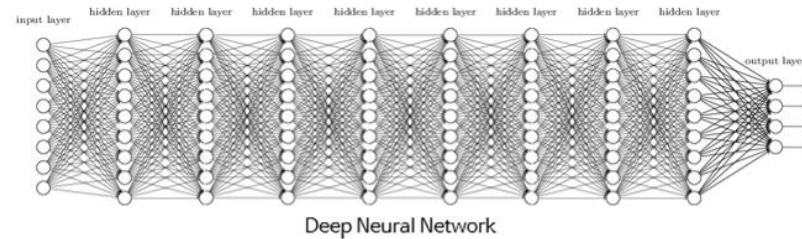
Задача классификации

- функция активации в выходном слое: сигмоида или softmax
- функция потерь: Cross Entropy или Правдоподобие

Затухающий градиент.

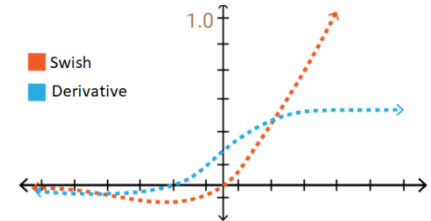
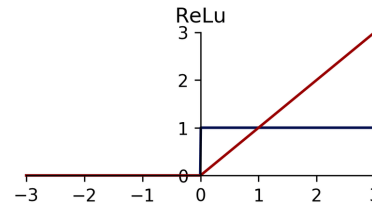
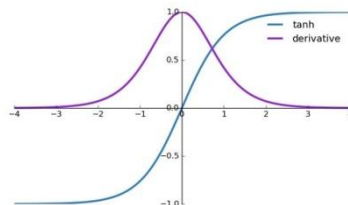
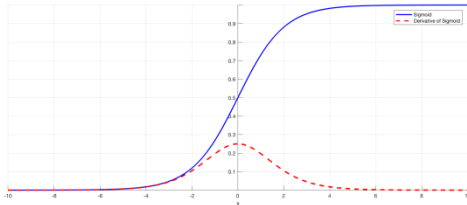
Затухающий градиент

- ❑ Уменьшение величины градиента с ростом числа слоев, пройденных в ходе обратного распространения ошибки.
- ❑ Малые значения производных функции активации.
- ❑ Градиент функции ошибки в промежуточном слое зависит от произведения всех производных функций активации в прошедших слоях.
- ❑ Коррекция весов в первых слоях сети существенно меньше чем в последних.



Функция активации

Activation	Function	Derivative	Min	Max
Logistic	$\sigma(z) = \frac{1}{1+e^{-z}}$	$\sigma(z)(1 - \sigma(z))$	$y \rightarrow 0$	0.25
Hyperbolic Tangent	$\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$	$1 - (\tanh(z))^2$	$y \rightarrow 0$	1
ReLU	$\max(0, z)$	$\begin{cases} 0, & z < 0 \\ 1, & z \geq 0 \end{cases}$	0	1
Leaky ReLU	$\max(0.2z, z)$	$\begin{cases} 0.2, & z < 0 \\ 1, & z \geq 0 \end{cases}$	0	1
Swish	$f(z) = z \cdot \sigma(z)$	$f(z) + \sigma(z)(1 - f(z))$	≈ -0.099	≈ 1.0998

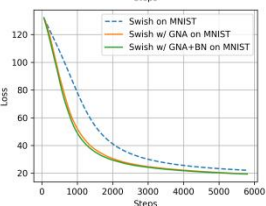
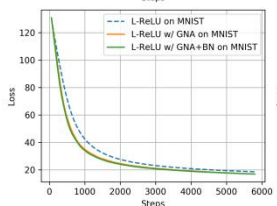
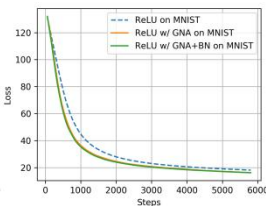
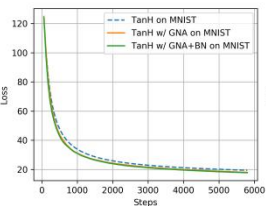
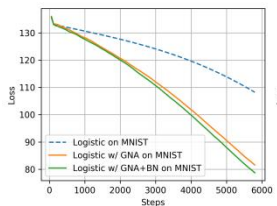
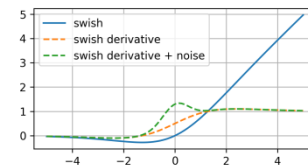
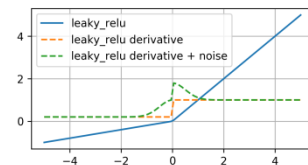
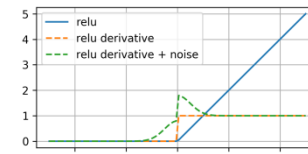
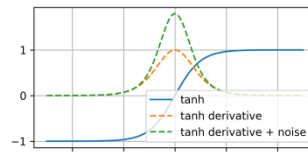
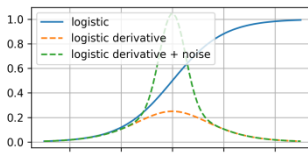


□ Применение активационных функций с большим диапазоном значений производной (ReLU, Swish, варианты ReLU) способствует решению проблемы затухающего градиента

Добавление «шума» в значение градиента

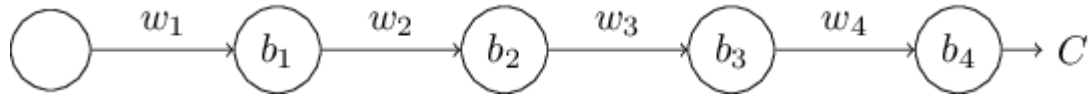
$$\nabla_{\theta_t} J := \nabla_{\theta_t} J + \mathcal{N}(0, \sigma_t^2)$$

$$\sigma_t^2 := \frac{\eta = 1}{(1 + t)^{\gamma=0.55}}$$



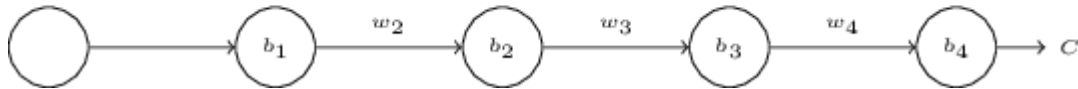
Глубина НС

- ❑ Глубокая НС → малый градиент на начальных слоях.
- ❑ 4 слоя с одним нейроном в каждом слое



- ❑ Градиент на первом слое

$$\frac{\partial C}{\partial b_1} = \sigma'(z_1) \times w_2 \times \sigma'(z_2) \times w_3 \times \sigma'(z_3) \times w_4 \times \sigma'(z_4) \times \frac{\partial C}{\partial a_4}$$

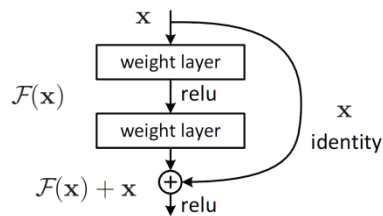


- ❑ При инициализации весов нормальным распределением с небольшим std → большинство $|w_j| < 1$, $\max(\sigma'(z)) = 0.25$ → $|w_j \times \sigma'(z)| < 0.25$ → $\partial C / \partial b_1 \ll \partial C / \partial b_3$

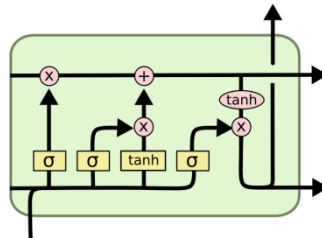
«Восстановление» затухающего градиента

□ Изменения в архитектуре НС:

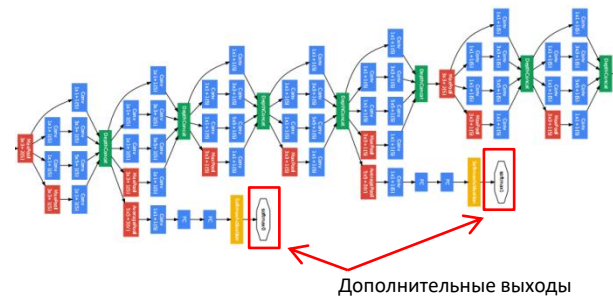
- Введение параллельных связей: обратное распространение градиента без умножения на производную функции активации: ResNet (остаточные связи), GRU и LSTM (передачи вектора состояния).
- Использование дополнительных выходов сети на этапе обучения: GoogLeNet



Блок ResNet с остаточной связью



LSTM Нейрон



Фрагмент GoogLeNet

Вопросы

- Основное отличие сигмоидальной функции от RELU, которое влияет на процесс обучения НС.
- Какие функции потерь предпочтительнее при обучении НС, решающих задачу регрессии, и в НС для задач классификации?
- Что такое проблема «затухающего градиента»?