Искусственные нейронные сети I

Сергей Николенко

Машинное обучение — ИТМО, осень 2006

Outline

Почему мы лучше?

- Компьютер считает быстрее человека
- Но гораздо хуже может:
  - понимать естественный язык
  - узнавать людей
  - обучаться в широком смысле этого слова
  - ...
- Почему так?

Строение мозга

Как человек всего этого добивается?

- В мозге много нейронов
- Но цепочка нейронов, которые успевают поучаствовать в принятии решения, не может быть длиннее нескольких сот штук!
- Значит, мозг очень хорошо структурирован в этом смысле

Искусственные нейронные сети

Основная мысль позаимствована у природы: есть связанные между собой нейроны, которые передают друг другу сигналы. Есть нейронные сети, которые стараются максимально точно моделировать головной мозг; это уже не AI и не наша тема.

Общая структура

- Есть сеть из нейронов, соединённых между собой
- Нейроны возбуждаются под действием входов и передают возбуждение (либо как один бит, либо с каким-то значением) дальше
- В результате последний нейрон на выход подаёт ответ

Как построить один нейрон?

Outline

Общие требования к модели

Суть модели:

- Нейрон возбуждается, если выполнено некоторое условие на входах;
- Затем он передаёт свой импульс дальше.

Нейрон возбуждается под действием какой-то функции от входов. Такая конструкция называется *перцептроном*. Это простейшая модель нейрона в искусственной нейронной сети.

Линейный перцептрон

У линейного перцептрона заданы:

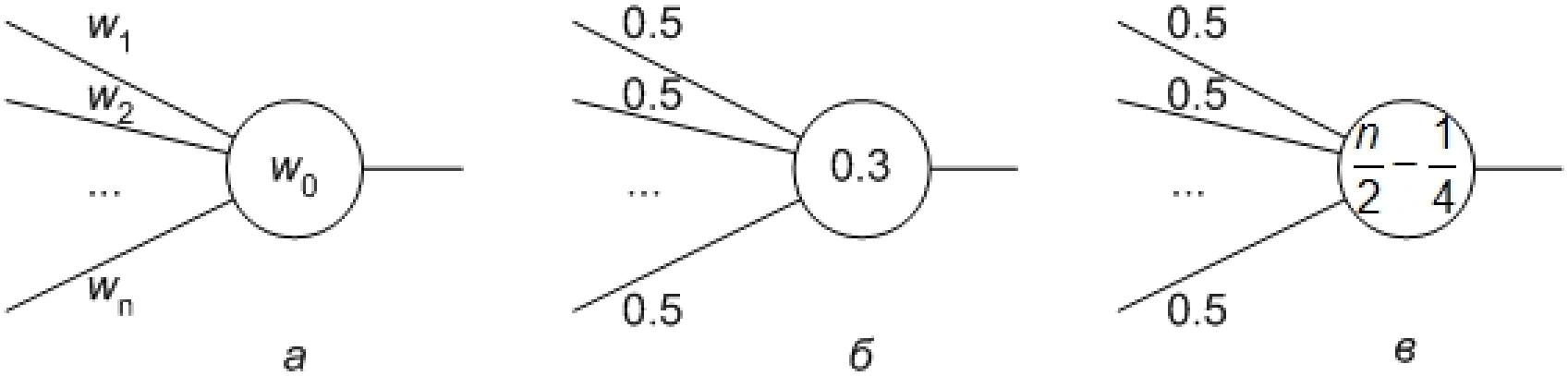
- n весов $w_1, w_2, \ldots, w_n$;
- лимит активации w_0;
- выход перцептрона $o(x_1, \ldots, x_n)$ вычисляется так:

$$o(x_1, \ldots, x_n) = \begin{cases} 1, \text{ если } w_0 + w_1 x_1 + \ldots + w_n x_n > 0, \\ -1 \text{ в противном случае.} \end{cases}$$

- или запишем иначе, введя переменную $x_0 = 1$:

$$o(x_1, \ldots, x_n) = \begin{cases} 1, \text{ если } \sum_i w_i x_i > 0, \\ -1 \text{ в противном случае.} \end{cases}$$

Примеры перцептронов



а *б* *в*

- *а* — общий вид перцептрона
- *б* — дизъюнкция
- *в* — конъюнкция

Сила перцептронов

- Один перцептрон может реализовать любую гиперплоскость, рассекающую пространство возможных решений. Иначе говоря, если прообразы 0 и 1 у целевой функции линейно отделимы, то *одного перцептрона достаточно*.
- Но он не может реализовать линейно неотделимое множество решений, например, XOR.
- А вот сеть из нескольких уровней перцептронов уже и XOR может.

Упражнение

Докажите, что любая булевская функция представима в виде построенной из перцептронов искусственной нейронной сети глубины 2.

Outline

Общие принципы

Как обучать перцептрон?

- Всё, что может у перцептрона меняться — это веса w_i, $i = 0..n$.
- Их мы и будем подправлять при обучении.
- Если перцептрон отработал правильно, веса не меняются.
- Если неправильно — сдвигаются в нужную сторону.

Perceptron training rule

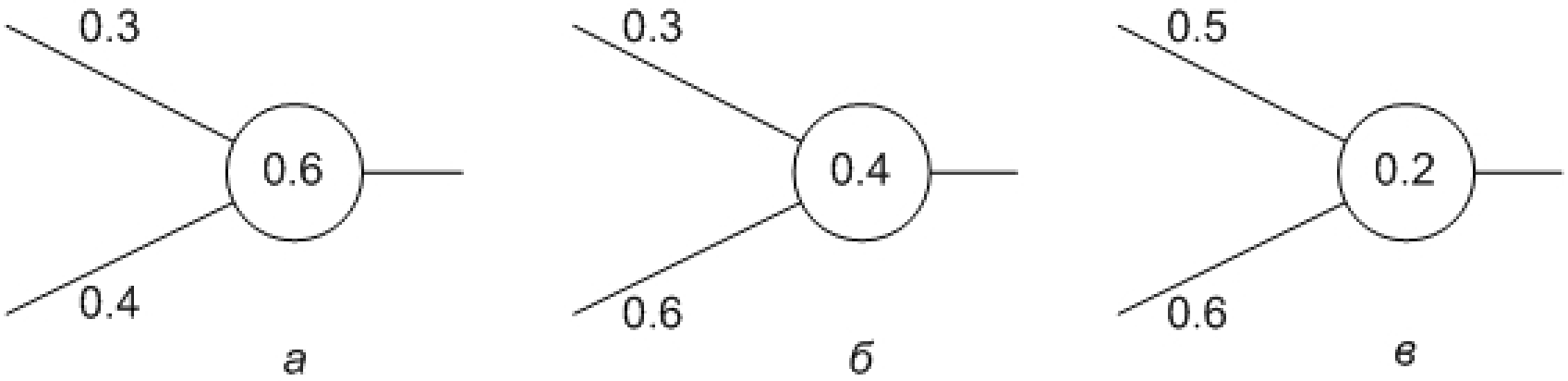
Простейшее правило:

$$w_i \leftarrow w_i + \eta(t - o)x_i,$$

где:

- t — значение целевой функции
- o — выход перцептрона
- $\eta > 0$ — небольшая константа (обычно 0.05–0.2), которая задаёт скорость обучения

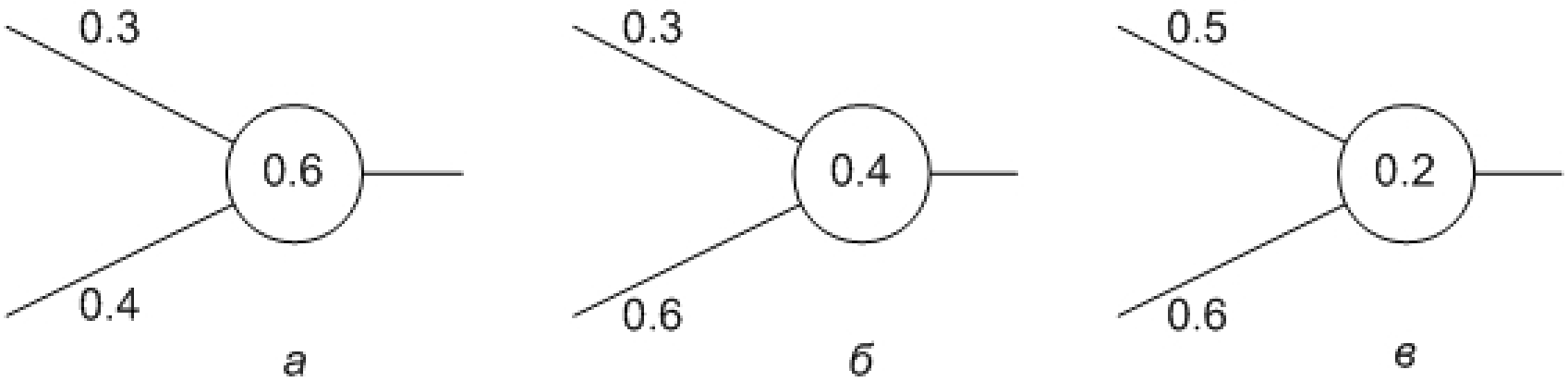
Пример обучения перцептрона



Мы хотим научить перцептрон распознавать дизъюнкцию.

- Рисунок *а* — перед началом обучения.
- Первый тест: $x_1 = 0,\ x_2 = 1 \Rightarrow t = 1$.
- Перцептрон тест не проходит.

Пример обучения перцептрона

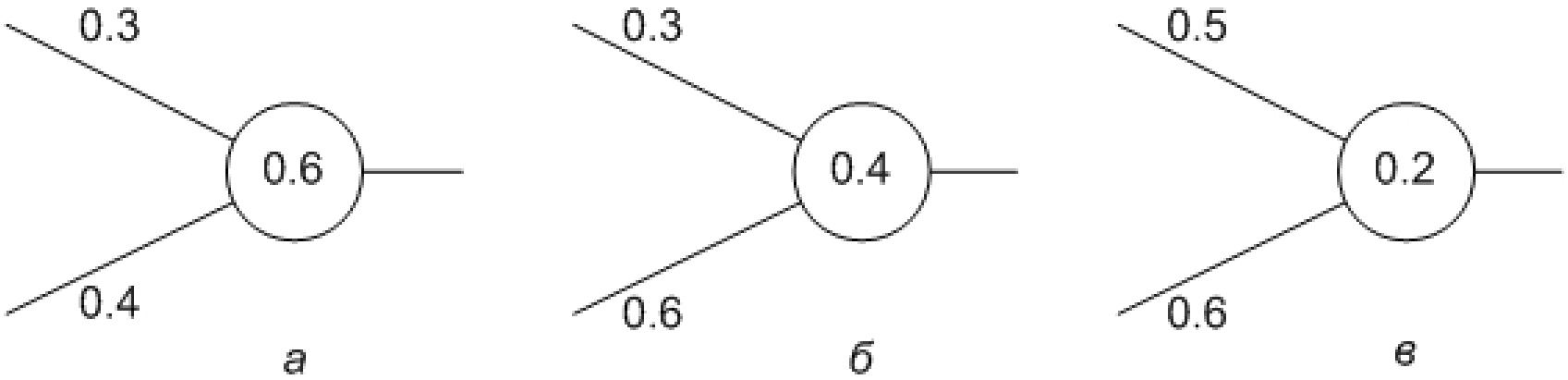


- Поправки на первом шаге:

$$
\begin{aligned}
w_0 &\leftarrow w_0 + \eta(t-0)x_0 = -0.6 + 0.1 \cdot (1-(-1)) \cdot 1 = -0.4,\\
w_1 &\leftarrow w_1 + \eta(t-0)x_1 = 0.3 + 0.1 \cdot (1-(-1)) \cdot 0 = 0.3,\\
w_2 &\leftarrow w_2 + \eta(t-0)x_2 = 0.4 + 0.1 \cdot (1-(-1)) \cdot 1 = 0.6.
\end{aligned}
$$

- После первого шага получаем перцептрон на рисунке *б*
- Второй тест: $x_1 = 1,\ x_2 = 0 \Rightarrow t = 1.$
- Перцептрон опять тест не проходит.

Пример обучения перцептрона

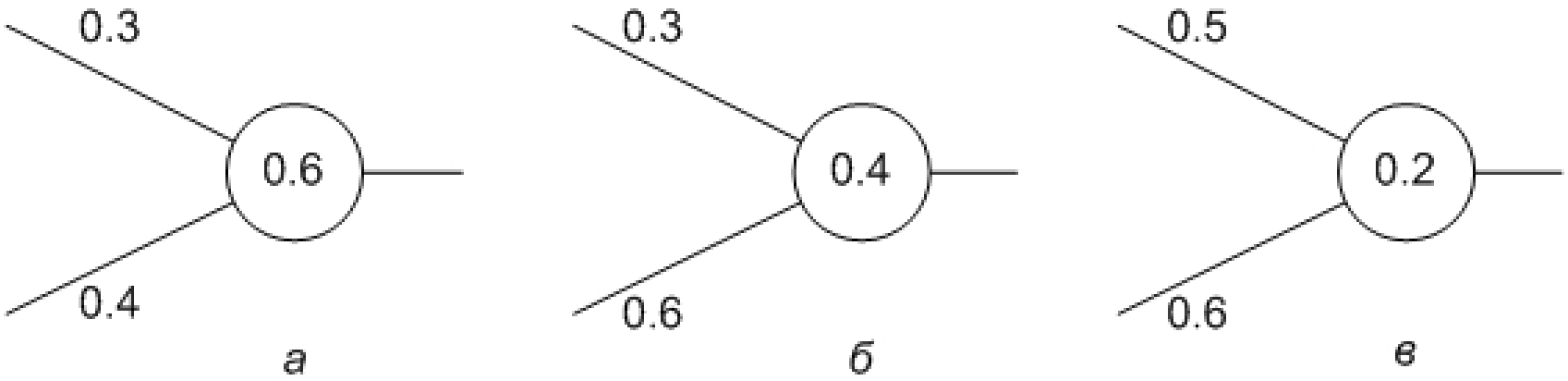


- Поправки на втором шаге:

$$
\begin{aligned}
w_0 &\leftarrow w_0 + \eta(t-0)x_0 = -0.4 + 0.1 \cdot (1-(-1)) \cdot 1 = -0.2, \\
w_1 &\leftarrow w_1 + \eta(t-0)x_1 = 0.3 + 0.1 \cdot (1-(-1)) \cdot 1 = 0.5, \\
w_2 &\leftarrow w_2 + \eta(t-0)x_2 = 0.6 + 0.1 \cdot (1-(-1)) \cdot 0 = 0.6.
\end{aligned}
$$

- Итого получается перцептрон с рисунка *в*. Он уже реализует дизъюнкцию правильно.

Пример обучения перцептрона



Если бы $\eta = 0.01$, веса были бы исправлены в нужную сторону, но недостаточно. Поэтому нужно запускать алгоритм по имеющимся тестовым примерам до тех пор, пока очередной прогон алгоритма по всем тестам не оставит все веса на месте.

Алгоритм обучения перцептрона

$\texttt{PerceptronTraining}(\eta, \{x_i^j, t^j\}_{i=1,j=1}^{n,m})$

- Инициализировать $\{w_i\}_{i=0}^{n}$ маленькими случайными значениями.
- $\texttt{WeightChanged} = \texttt{true}$.
- Пока $\texttt{WeightChanged} = \texttt{true}$:
  - $\texttt{WeightChanged} = \texttt{false}$.
  - Для всех j от 1 до m:
    - Вычислить

$$o^j = \begin{cases} 1, \text{ если } w_0 + w_1 x_1^j + \ldots + w_n x_n^j > 0, \\ -1 \text{ в противном случае.} \end{cases}$$

    - Если $o^j \neq t^j$:
      — $\texttt{WeightChanged} = \texttt{true}$.
      — Для каждого i от 0 до n изменить значение w_i по правилу

$$w_i \leftarrow w_i + \eta(t^j - o^j)x_i^j.$$

- Выдать значения $w_0, w_1, \ldots, w_n$.

Обучение перцептрона на Python

```
def PerceptronTraining(eta,x):
    import random
    w=[]
    for i in range(len(x[0])):
        w.append((random.randrange(-5,5))/50.0)
    WeightsChanged=True
    while (WeightsChanged==True):
        WeightsChanged=False
        for xj in x:
            t,o,curx=xj[0],0,[1]+xj[1:len(xj)]
            for i in xrange(len(w)): o+=w[i]*curx[i]
            if o>0: o=1
            else: o=-1
            if (o==t): continue
            WeightsChanged=True
            for i in xrange(len(w)): w[i]+=eta*(t-o)*curx[i]
    return w
```

Сходимость

Этот алгоритм сходится всегда, когда это возможно.

Теорема

Если конечное множество точек $C_1 \subset \{0,1\}^n$ *можно в* $\{0,1\}^n$ *отделить гиперплоскостью от конечного множества точек* $C_2 \subset \{0,1\}^n$, *то алгоритм обучения перцептрона за конечное количество шагов выдаёт параметры перцептрона, который успешно разделяет множества* C_1 *и* C_2.

Спасибо за внимание!

- Lecture notes, слайды и коды программ появятся на моей homepage:

  `http://logic.pdmi.ras.ru/~sergey/index.php?page=teaching`

- Присылайте любые замечания, коды программ на других языках, решения упражнений, новые численные примеры и прочее по адресам:
  `sergey@logic.pdmi.ras.ru`, `smartnik@inbox.ru`