искусственные нейронные сети

Евгений Борисов

Нейросети

нейронные сети

- вычислительная нейробиология

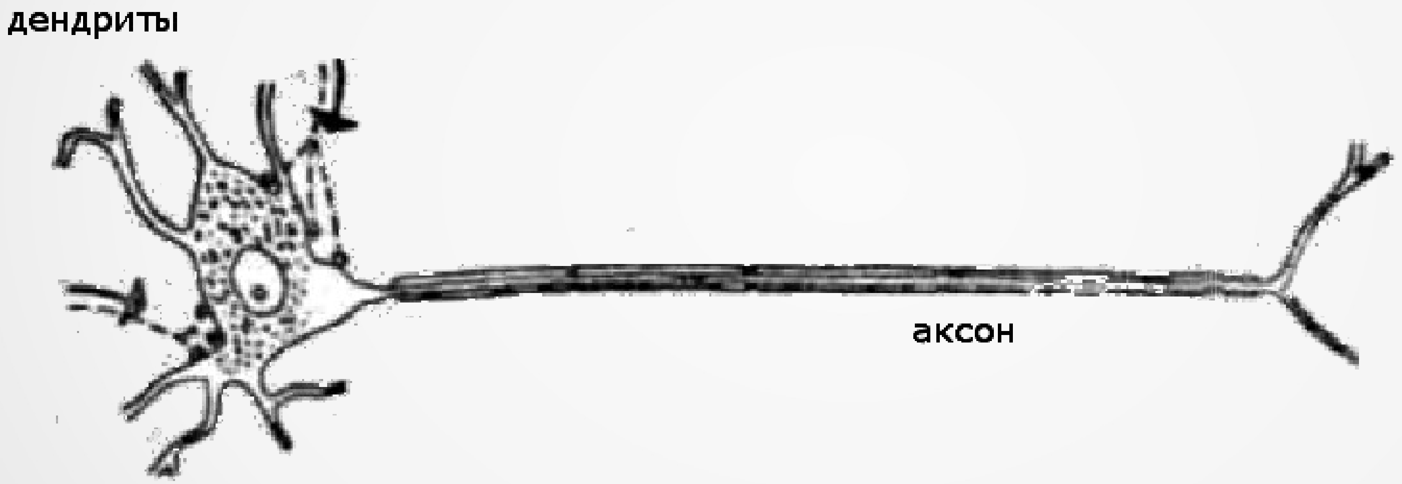
  цель: моделировать процессы в живых организмах

- теория искусственных нейронных сетей

  цель: построить искусственную интеллектуальную систему

Нейросети

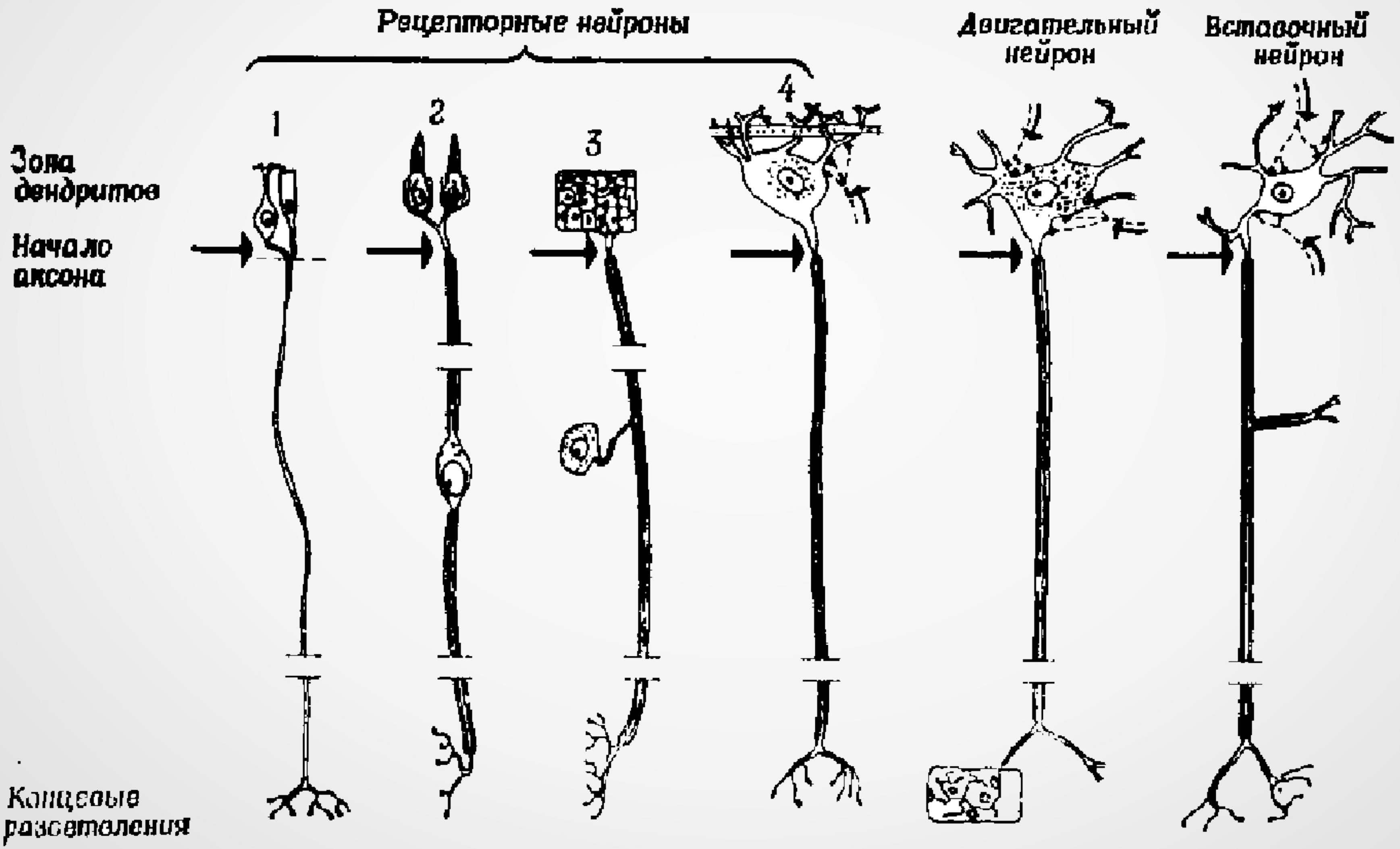
нервная клетка



Вилли К., Детье В.   Биология, М.: Мир, 1975

Нейросети

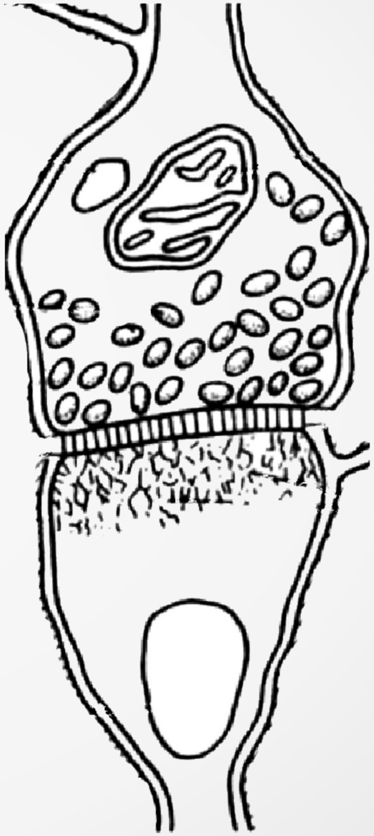
различные типы нервных клеток



Вилли К., Детье В. Биология, М.: Мир, 1975

Нейросети

синапс - межнейронные соединения



ширина зазора ~50нм

однонаправленная передача сигнала

вещество-нейромедиатор передаёт сигнал химическим способом

Вилли К., Детье В. Биология, М.: Мир, 1975

Нейросети

нервный импульс - электрохимическая реакция

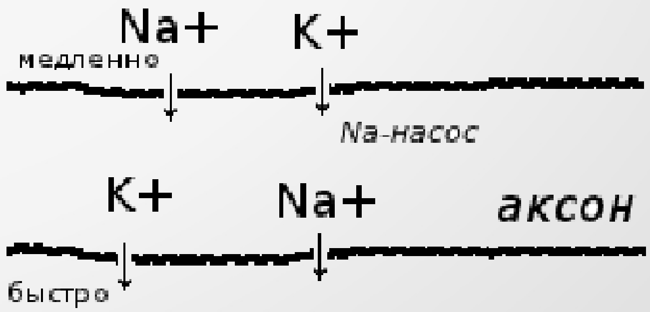
мембранная теория

разница потенциалов на клеточной мембране ~60мВ

при стимуляции разряжается, выбрасывает нейромедиатор

изменяемая проницаемость мембраны

проникновение инонов Na+ K+ через мембрану
с разной скоростью образует разницу потенциалов



Вилли К., Детье В.   Биология, М.: Мир, 1975

Нейросети

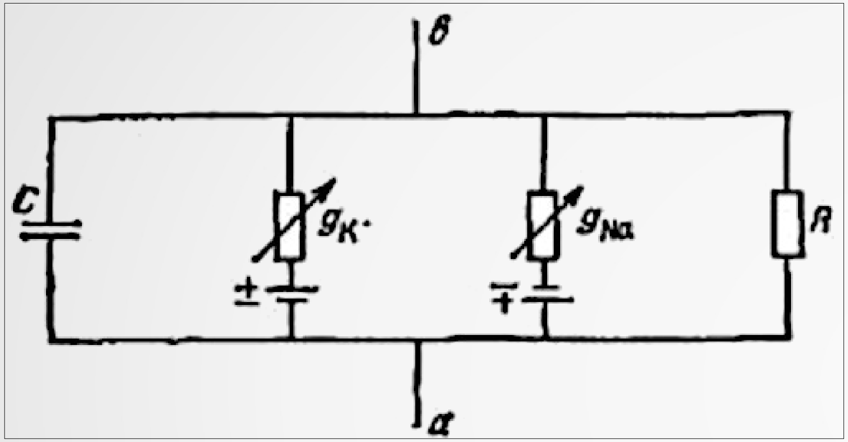
модель Ходжкина-Хаксли

Hodgkin, A., and Huxley, A. Currents carried by sodium and potassium ions through the membrane of the giant axon of loligo. - J. Physiol. (I952) II6, 449-472

Laboratory of the Marine Biological Association, Plymouth
Physiological Laboratory, University of Cambridge

Нейросети

модель Ходжкина-Хаксли изменение проницаемости при сдвиге потенциала на мембране нервной клетки



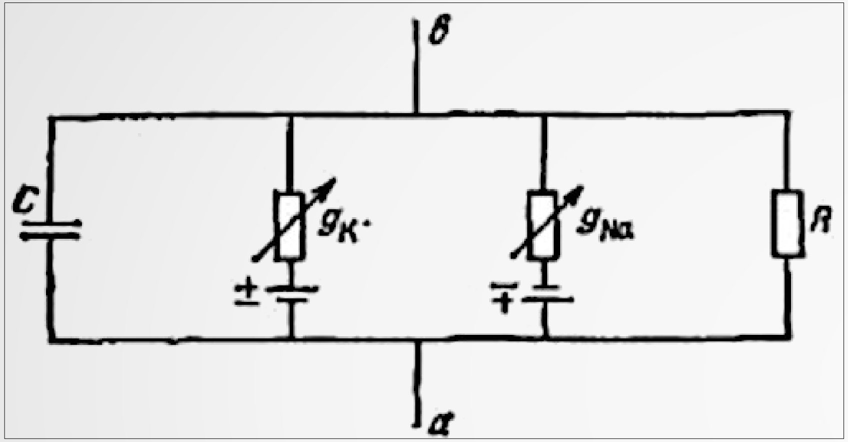
эквивалентная схема мембраны аксона кальмара

в - внешняя среда (вода)
а - внутреняя среда (аксоплазма)

параллельно включенные
- емкость С
- два элемента-источника тока
- переменные сопротивления определяются калиевой gK и gNa натриевой проводимостями

Нейросети

модель Ходжкина-Хаксли изменение проницаемости при сдвиге потенциала на мембране нервной клетки



$$C\frac{dV}{dt} = g_K(V - V_K) + g_{Na}(V - V_{Na}) + I(t),$$

$$g_K = g_{K\,max}\cdot n^4,$$

$$g_{Na} = g_{Na\,max}\cdot m^3 h,$$

$$\frac{dn}{dt} = \alpha_n(1-n) - \beta_n\cdot n,$$

$$\frac{dm}{dt} = \alpha_m(1-m) - \beta_m\cdot m,$$

$$\frac{dh}{dt} = \alpha_h(1-h) - \beta_h\cdot h,$$

$$\alpha_n = \frac{0{,}01\,(V-10)}{1-e^{(10-V)/10}}, \qquad \beta_n = 0{,}125e^{-V/80},$$

$$\alpha_m = \frac{0{,}1\,(V-25)}{1-e^{(25-V)/10}}, \qquad \beta_m = 4e^{-V/18},$$

$$\alpha_h = 0{,}7e^{-V/20}, \qquad \beta_h = \frac{1}{1+e^{(30-V)/10}}.$$

эквивалентная схема мембраны аксона кальмара

в - внешняя среда (вода)
а - внутреняя среда (аксоплазма)

параллельно включенные
- емкость C
- два элемента-источника тока
- переменные сопротивления определяются калиевой gK и gNa натриевой проводимостями

Нейросети

Импульсная нейронная сеть

Pulsed neural networks, PNN
Спайковая нейронная сеть
Spiking neural network, SNN

сеть получает на входы серию импульсов
и выдаёт импульсы на выходе.

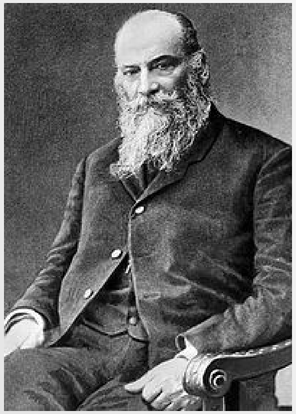
параметры связей импульсного нейрона
- время задержки и величина веса

Нейросети

про историю авиации ...

Нейросети

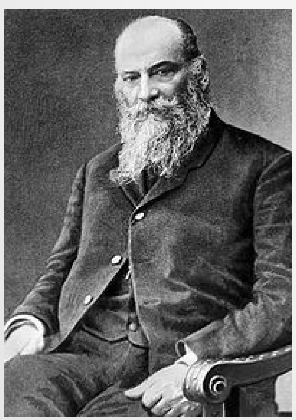
про историю авиации ...



Николай Егорович
Жуковский

Нейросети

про историю авиации ...



Николай Егорович
Жуковский

не копировать
но использовать идею

Нейросети

История развития искусственных нейронных сетей

[1950] математические модели биологического нейрона

McCulloch W.S., Pitts W. A logical Calculus of Ideas Immanent in Nervous Activity
– Bull. Mathematical Biophysics, 1943

[1960] модели нейронных сетей с одним обрабатывающим слоем

F.Rosenblatt Principles of Neurodinamics. - New York: Spartan Books,1962.

[1969] критика модели персептрона Розенблата

M.Minsky,S.Papert Perceptrons - an introduction to computational geometry, 1969, ISBN 0262130432

[1970-80] метод обратного распространения,
нейронные сети с несколькими обрабатывающими слоями

Галушкин А. И. Синтез многослойных систем распознавания образов.
– М.: «Энергия», 1974.

D.E.Rumelhart, G.E.Hinton, R.J.Williams Learning internal representations
by error propagation. // In Parallel distributed processing,
vol. 1, pp. 318-62. Cambridg, MA: MIT Press, 1986.

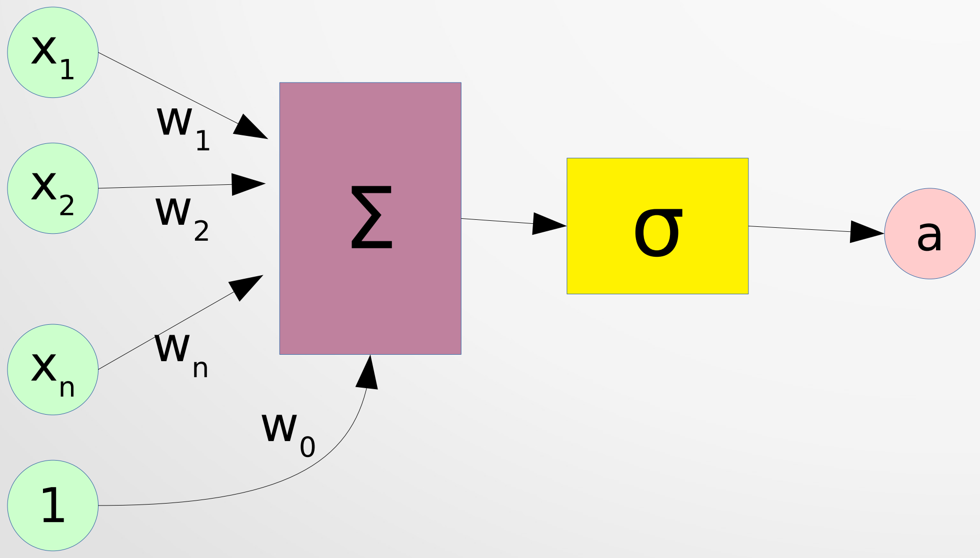
[2005] концепция Deep Learning

Нейросети

модель МакКаллока-Питтса (1943)

$$a(x,w)=\sigma\left(\sum_{i=1}^{n} x_i \cdot w_i - w_0\right)=\sigma(\langle x,w\rangle)$$

$\mathbf{x_i}$ - вход
$\mathbf{w_i}$ - вес связи
$\boldsymbol{\sigma}$ - функция активации

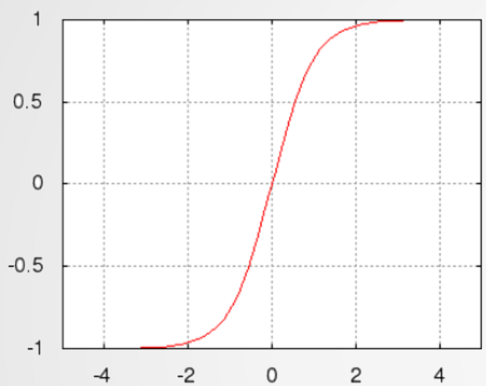


состояние нейрона

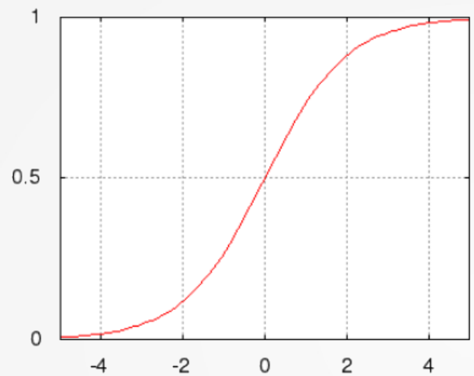
$$s(x,w)=\sum_{i=1}^{n} x_i \cdot w_i - w_0$$

Нейросети

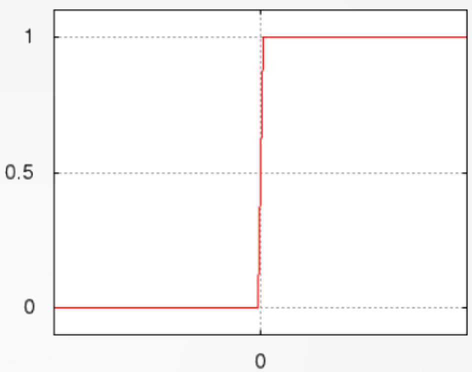
примеры функций активации



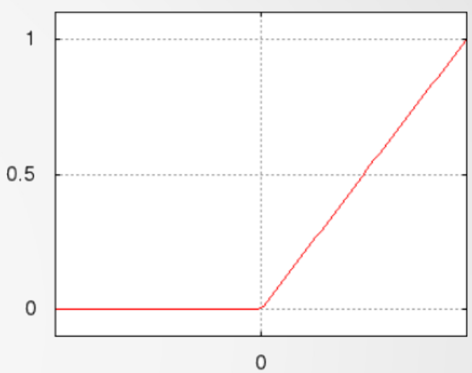
биполярный сигмоид



экспоненциальный сигмоид



пороговая функция



кусочно-линейная ReLU

softmax (экспоненциальная нормализация)
выходного слоя

$$(y_1, \ldots, y_m) = softmax(s_1, \ldots, s_m) = \frac{\exp(s)}{\sum_j \exp(s_j)}$$

стохастическая, выход нейрона
с вероятностью p равен 1
и (1-p) равен 0

$$p = \frac{1}{1 + \exp(-s)}$$

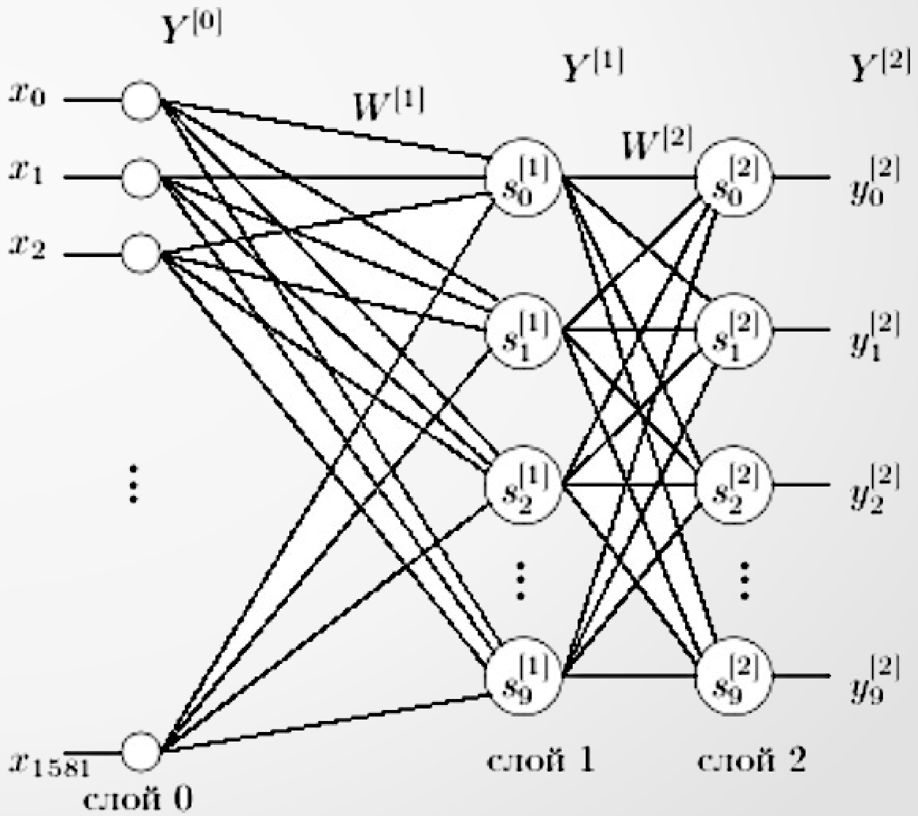
Нейросети

коннекционизм -
модель ИИ из связанных между собой простых элементов

многослойная сеть прямого распространения

нейроны объединены в слои

сигнал распространяется послойно



Нейросети

коннекционизм -
модель ИИ из связанных между собой простых элементов

многослойная сеть прямого распространения

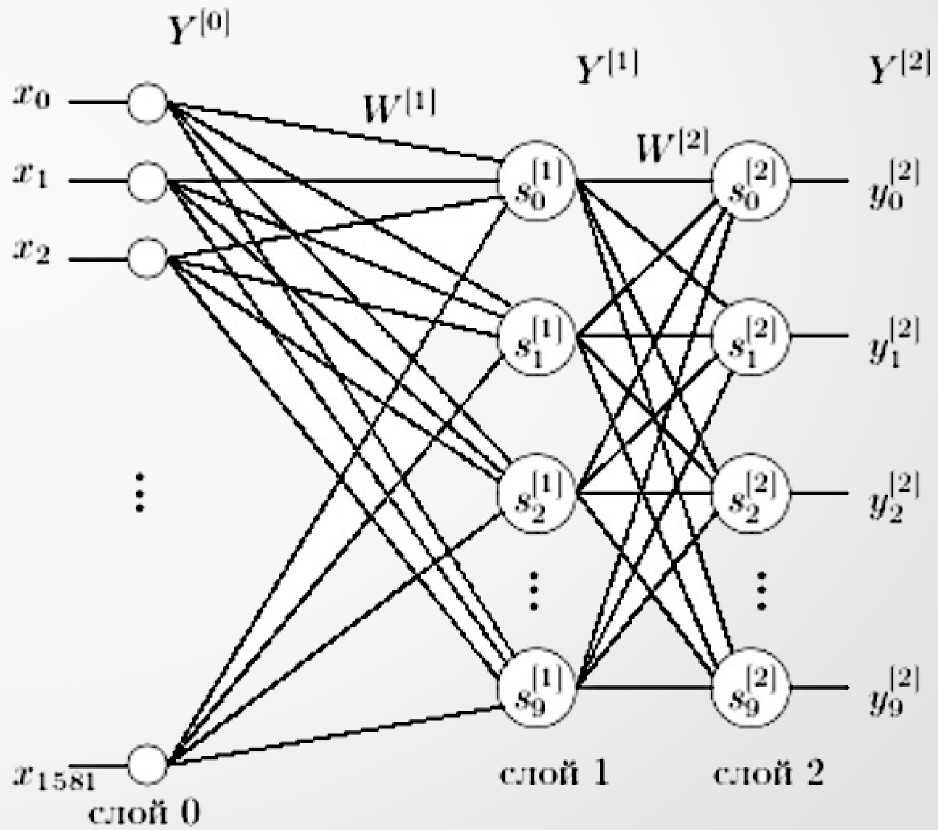
нейроны объединены в слои

сигнал распространяется послойно

входной рапределительный слой

обрабатывающие скрытые слои

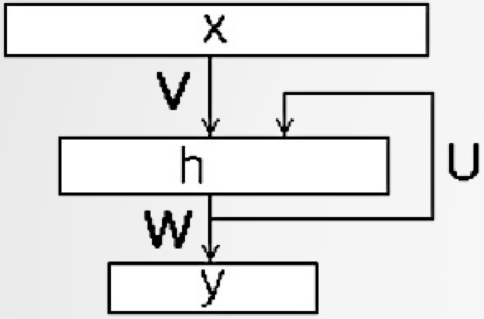
обрабатывающий выходной слой



Нейросети

другие типы моделей нейросетей

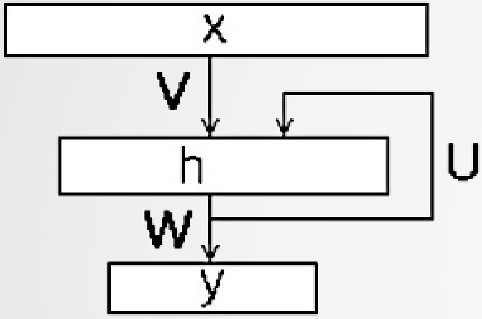
рекуррентные - Элман, LSTM



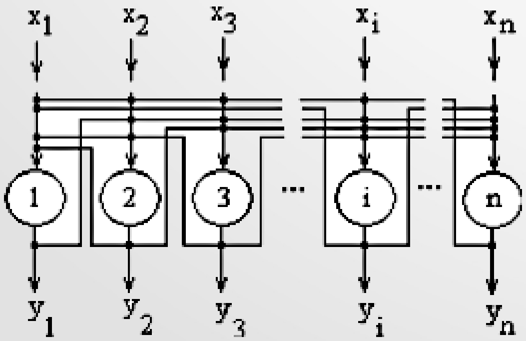
Нейросети

другие типы моделей нейросетей

рекуррентные - Элман, LSTM



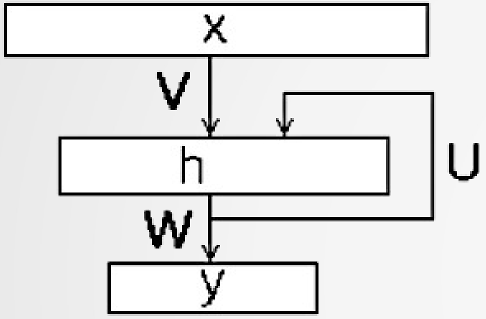
релаксационные - Хопфилд



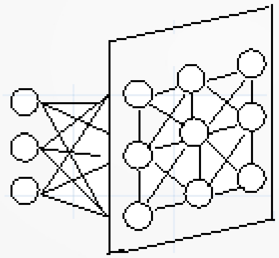
Нейросети

другие типы моделей нейросетей

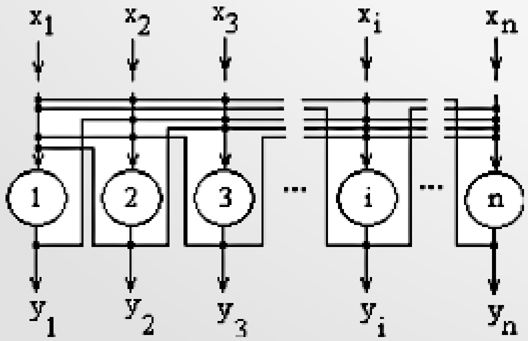
рекуррентные - Элман, LSTM



соревновательные - Кохонен



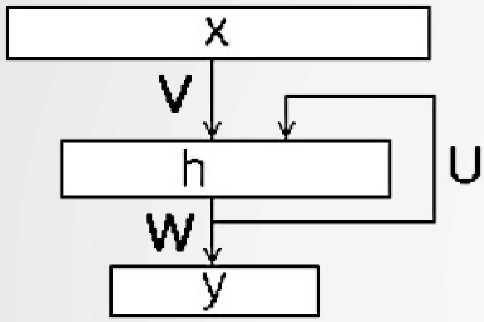
релаксационные - Хопфилд



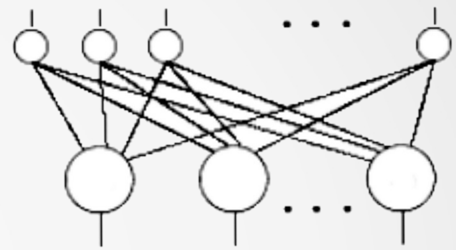
Нейросети

другие типы моделей нейросетей

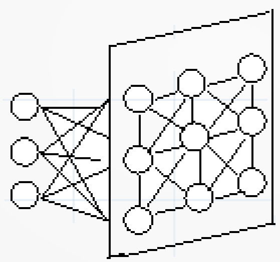
рекуррентные - Элман, LSTM



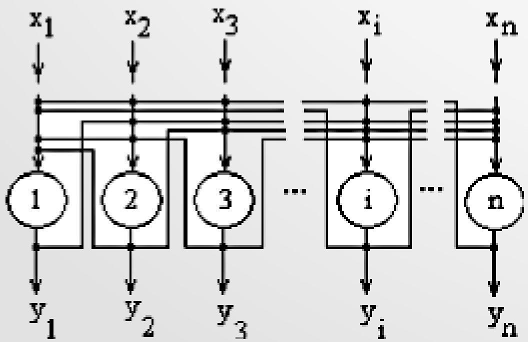
двунаправленные - Коско



соревновательные - Кохонен



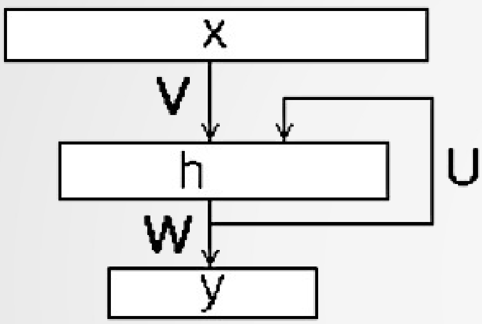
релаксационные - Хопфилд



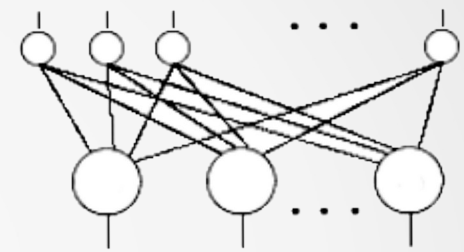
Нейросети

другие типы моделей нейросетей

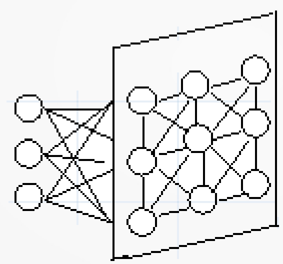
рекуррентные - Элман, LSTM



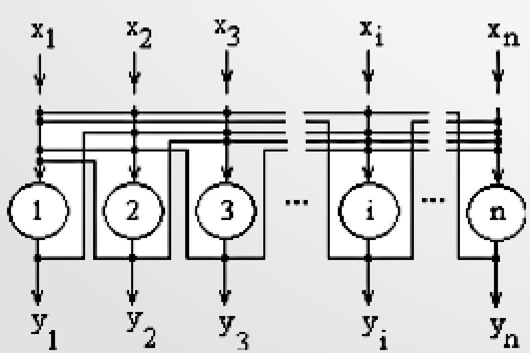
двунаправленные - Коско



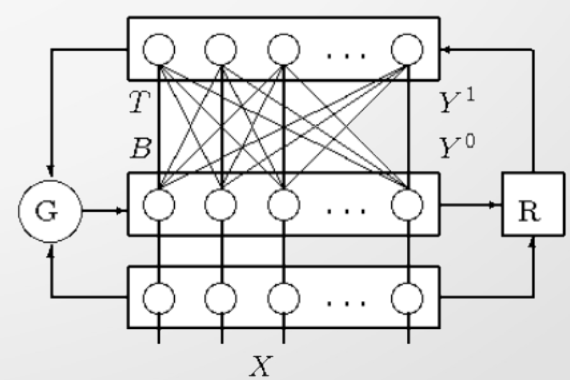
соревновательные - Кохонен



релаксационные - Хопфилд



адаптивный резонанс - Гроссберг



Нейросети

ИНС прямого распространения (MLP)

<u>о количестве обрабатывающих слоёв</u>

1 слой - гиперплоскость (линейный классификатор)

2 слоя - выпуклая разделяющая поверхность

3 слоя - поверхность любой формы

Нейросети

ИНС прямого распространения (MLP)

о количестве обрабатывающих слоёв

1 слой - гиперплоскость (линейный классификатор)

2 слоя - выпуклая разделяющая поверхность

3 слоя - поверхность любой формы

Теоретическое обоснование

Колмогоров А.Н. О представлении непрерывных функций нескольких переменных в виде суперпозиций непрерывных функций одного переменного и сложения // Докл. АН СССР, том 114, с. 953-956, 1957.

Арнольд В.И. О функциях трех переменных // Докл. АН СССР, том 114, N 4, 1957.

Hecht-Nielsen R. Kolmogorov's Mapping Neural Network Existence Theorem // IEEE First Annual Int. Conf. on Neural Networks, San Diego, 1987, Vol. 3, pp. 11-13.

Нейросети

ИНС прямого распространения (MLP)

о количестве обрабатывающих слоёв

1 слой - гиперплоскость (линейный классификатор)

2 слоя - выпуклая разделяющая поверхность

3 слоя - поверхность любой формы

многослойная нейросеть с линейной функцией активации эквивалентна однослойной

Нейросети

многослойная нейросеть прямого распространения с линейной функцией активации эквивалентна однослойной

Доказательство:

$$Y = a\left(a\left(a\left(X \cdot W_1\right) \cdot W_2\right) \ldots W_n\right)$$

если активация линейная   a(s)=s
то

$$Y = X \cdot W_1 \cdot W_2 \ldots W_n = X \cdot \left(W_1 \cdot W_2 \ldots W_n\right) = X \cdot W$$

Нейросети

ИНС прямого распространения (MLP)

три слоя - поверхность любой формы

получается, что строить нейросети
глубже трёх слоёв не имеет смысла ??...

«есть один нюанс...» (с)   :)

Нейросети

Про обучение многослойных нейросетей

$h: X \times W \rightarrow Y$ классификатор (X вход, W параметры, Y ответ)

$E: Y \times C \rightarrow \mathbb{R}$ функция потери (Y ответ, C класс)

обучение классификатора как задача оптимизации

$$E\left(h\left(X, W\right), C\right) \rightarrow \min_{W}$$

Нейросети

примеры функций потери

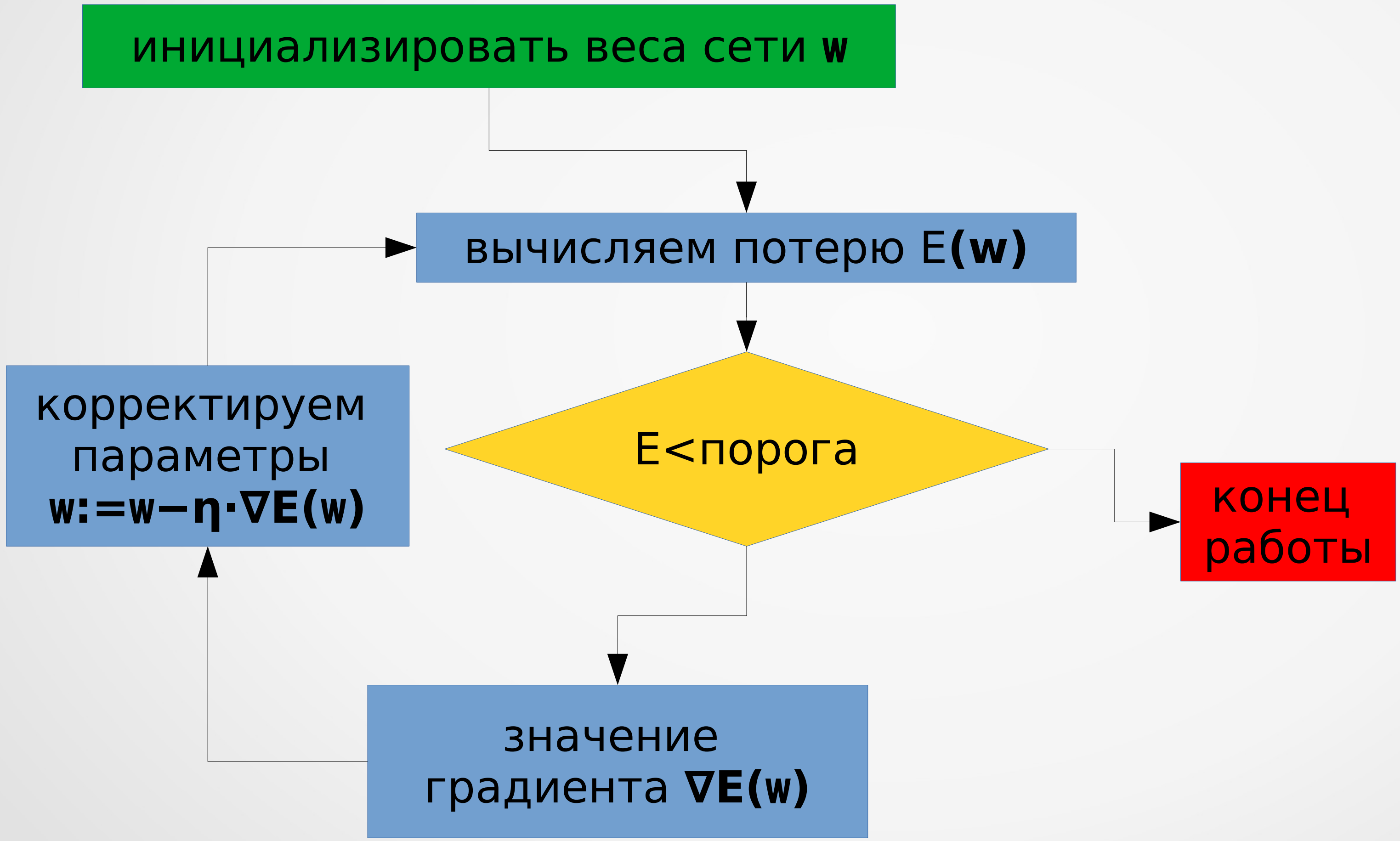
MSQE среднеквадратичное отклонение

Кросс-энтропия

Расстояние Кульбака-Лейблера

Нейросети

градиентный спуск (GD)



Нейросети

метод обратного распространения ошибки

вычисление градиента функции потери
для многослойной нейросети

$$\nabla E(W) = \left[\frac{\partial E}{\partial w_1}\dots,\frac{\partial E}{\partial w_k}\right]$$

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial y_j}\frac{\partial y_j}{\partial s_j}\frac{\partial s_j}{\partial w_{ij}}$$ градиент функции потери для ИНС

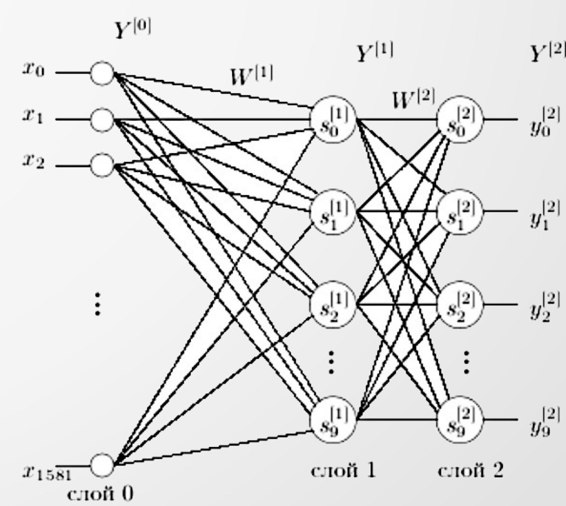
$$\frac{\partial s_j}{\partial w_{ij}}$$ выход i-того нейрона предыдущего слоя (определен явно)

$$\frac{\partial y_j}{\partial s_j}$$ производная активационной функции (можем вычислить)

$$\frac{\partial E_j}{\partial y_j}$$ ошибка нейрона номер j (определена для выходного слоя)

$$\delta_i := \frac{\partial E}{\partial y_i}$$ ошибка нейрона номер j для выходного слоя

$$\delta_i := \frac{\partial y_i}{\partial s_i}\cdot\sum_j \delta_j w_{ij}$$ ошибка нейрона номер j для скрытого слоя

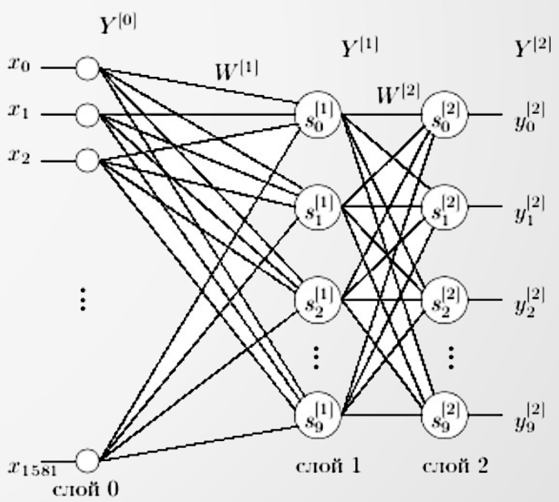


Нейросети

метод обратного распространения ошибки backProp

1. прямой проход:
   вычислить состояния нейронов s для всех слоёв и выход сети y

2. вычисляем значения ошибки выходного слоя $\delta := \partial E / \partial y$



3. обратный проход:
   последовательно от конца к началу
   вычисляем δ для всех скрытых слоёв

4. для каждого слоя вычисляем значение градиента
   $\nabla E = \partial E / \partial w = y \cdot \delta^T$

Нейросети

стратегии обучения

full batch - на каждой итерации используем все примеры

stochastic - на каждой итерации используем один случайный пример

mini batch - на каждой итерации используем случайное подмножество примеров

Нейросети

модификации градиентного спуска

момент или «тяжёлый шарик», вытаскивает из локальных минимумов

$$\Delta W_t := \eta \nabla E + \mu \Delta W_{t-1}$$

регуляризация - штрафует за чрезмерный рост весов
помогает бороться с переобучением

$$\Delta W_t := \eta (\nabla E + \rho W_{t-1}) + \mu \Delta W_{t-1}$$

Нейросети

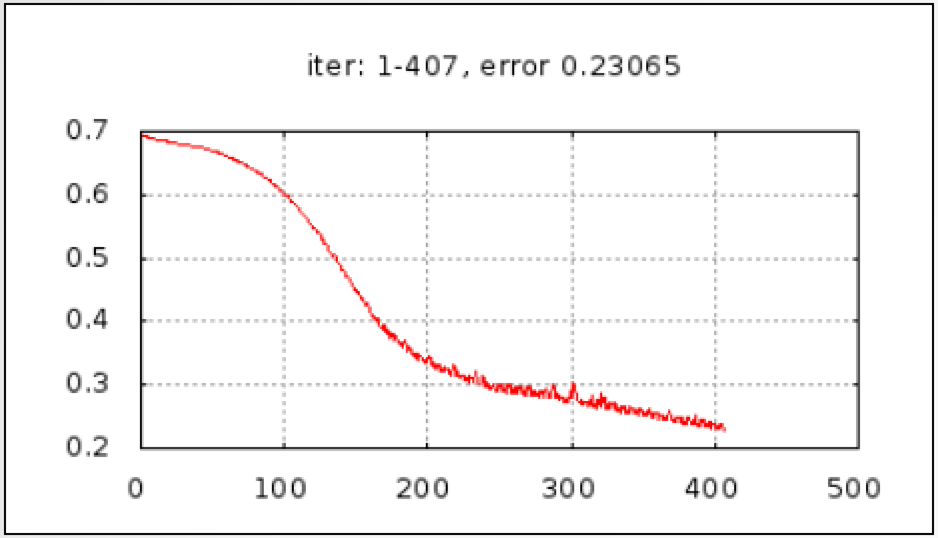


Рис.: история изменения ошибки ч.1

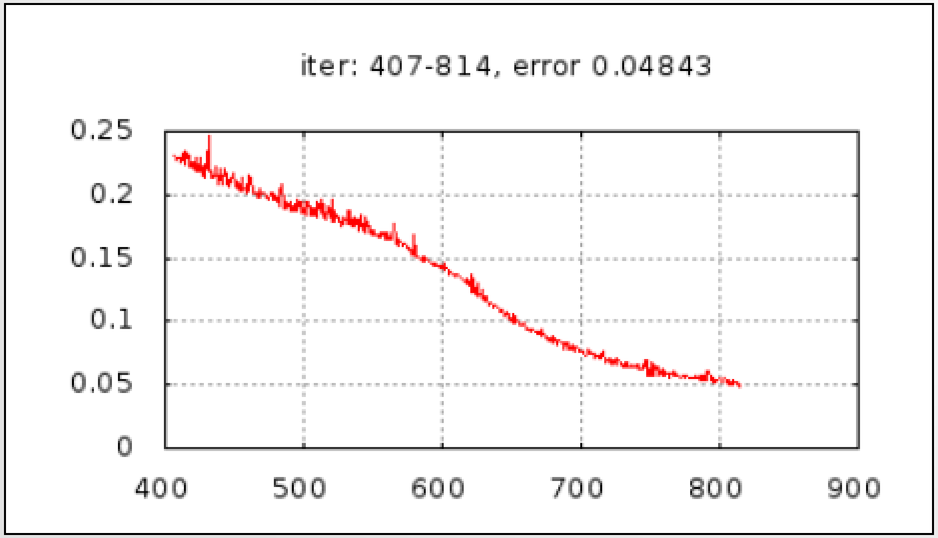


Рис.:история изменения ошибки ч.2

простой градиентный спуск

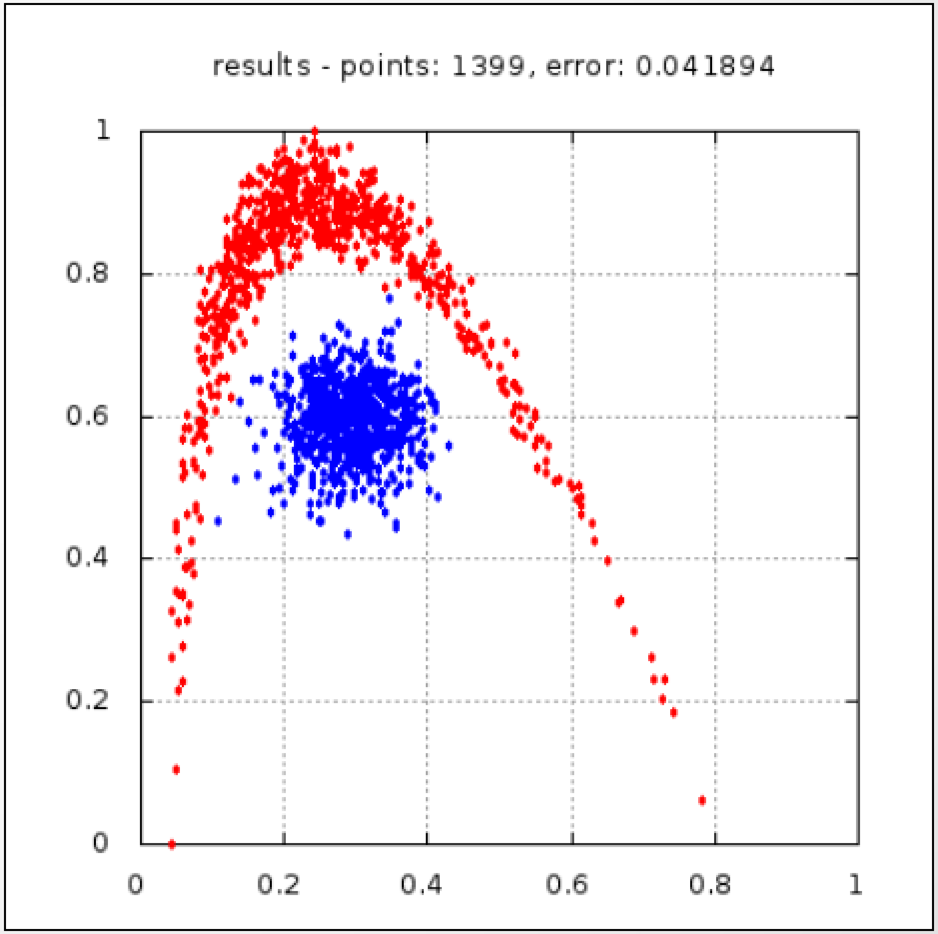


Рис.: результат теста

Нейросети

простой градиентный спуск

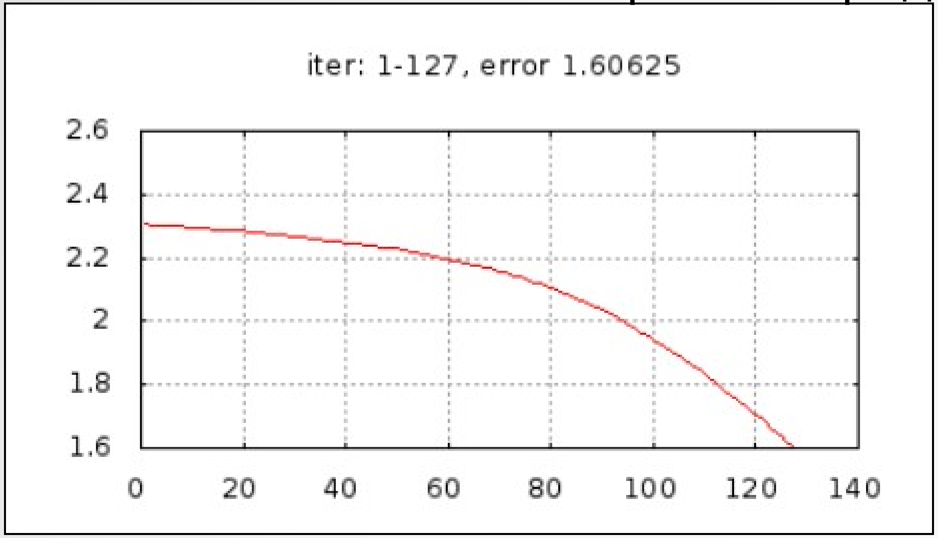


Рис.: история изменения ошибки ч.1

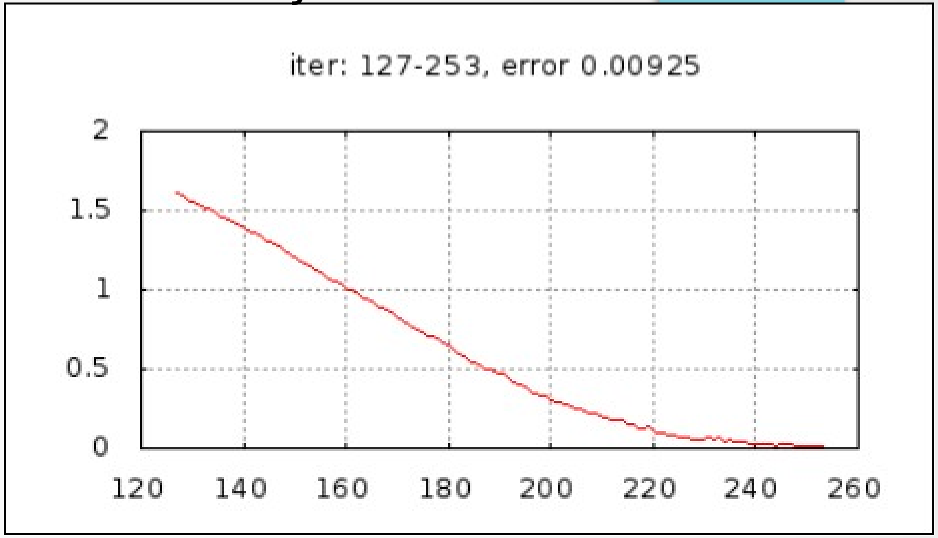


Рис.:история изменения ошибки ч.2

Рис.: состояния весов первого слоя

Нейросети

модификации градиентного спуска **quickProp**

параметр момента μ и коэффициент скорости обучения η задаются индивидуально для каждого параметра

$$\Delta W_t := \eta(\nabla E + \rho W_{t-1}) + \mu \Delta W_{t-1}$$

$$\eta = \begin{cases} \eta_0 & : \quad (\Delta W = 0) \vee (-\Delta W \cdot S > 0) \\ 0 & : \quad - \end{cases}$$

где $\eta_0 \in (0.01, 0.6)$ - константа, $S = \nabla E + \rho W$

Параметр момента выглядит следующим образом.

$$\mu = \begin{cases} \mu_{max} & : \quad (\beta > \mu_{max}) \vee (\gamma < 0) \\ \beta & : \quad - \end{cases}$$

где $\mu_{max} = 1.75$ - константа,
$S = \nabla E + \rho W$,
$\beta = S(t)/(S(t-1) - S(t))$
$\gamma = S \cdot (-\Delta W) \cdot \beta$

Нейросети

quickProp

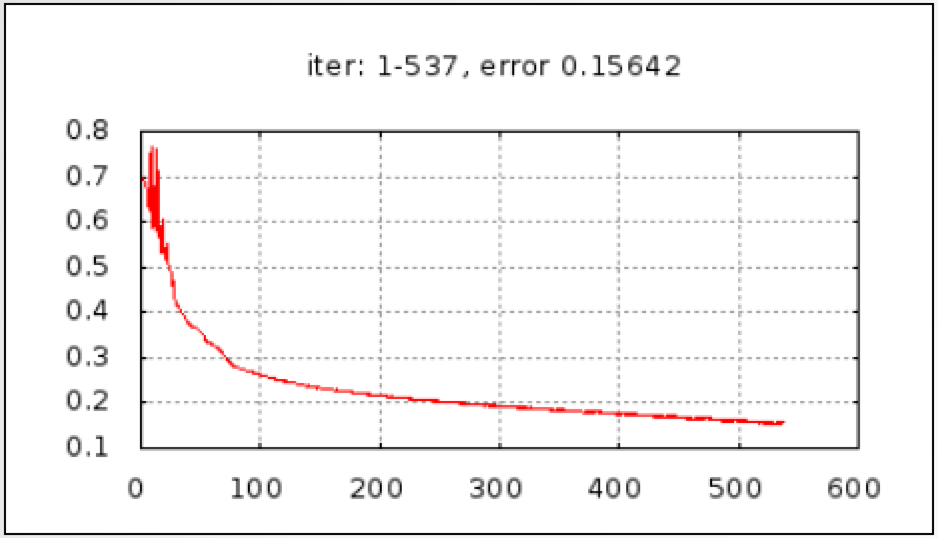


Рис.: история изменения ошибки ч.1

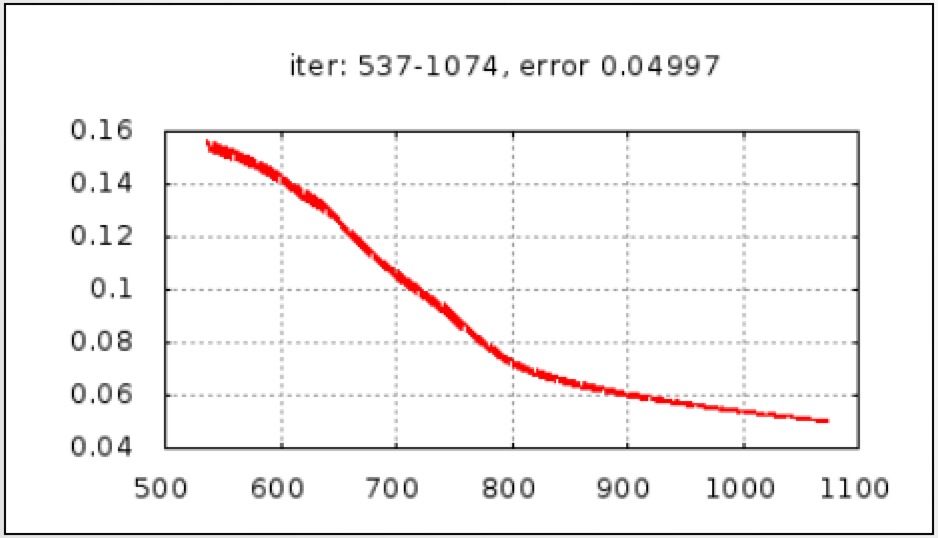


Рис.:история изменения ошибки ч.2

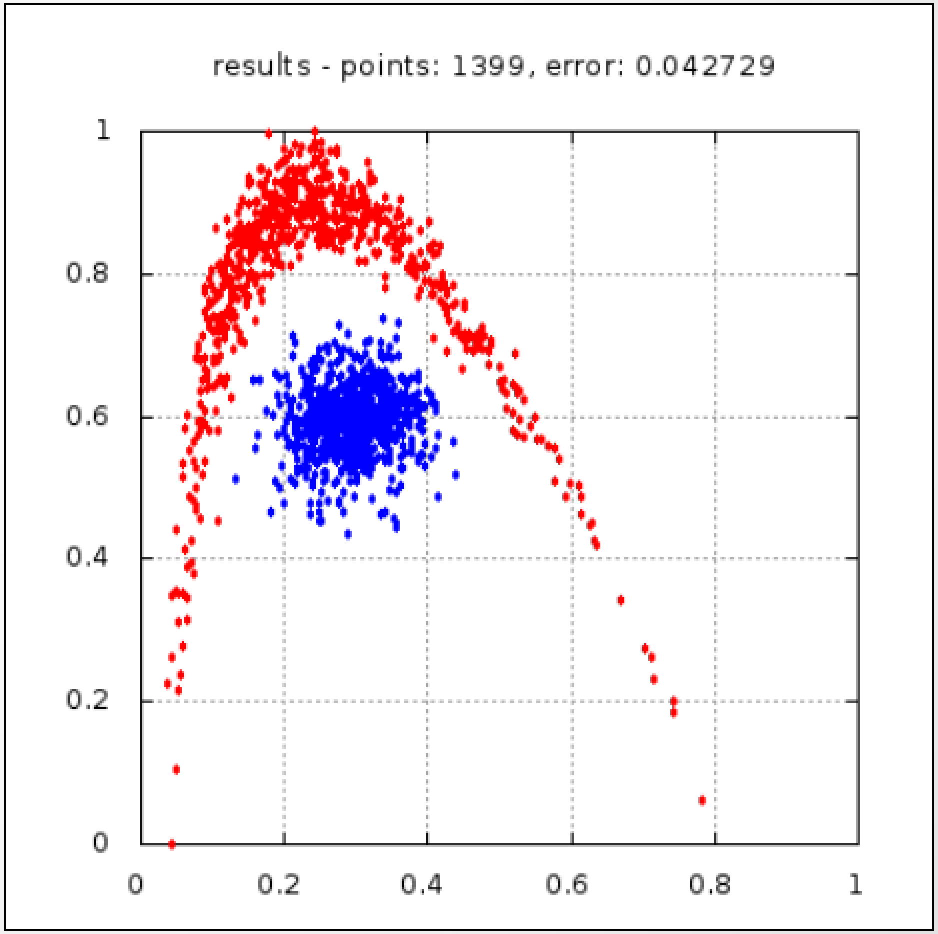


Рис.: результат теста

Нейросети

quickProp

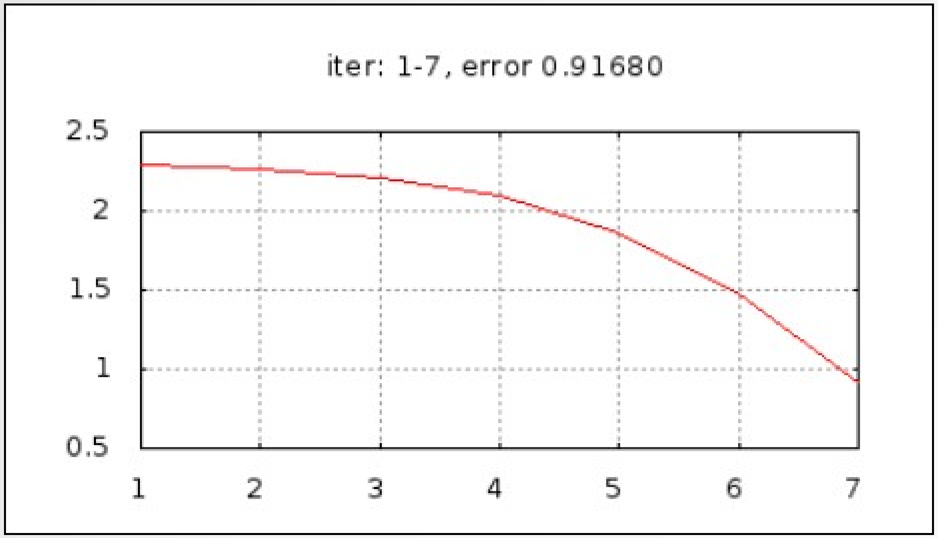


Рис.: история изменения ошибки ч.1

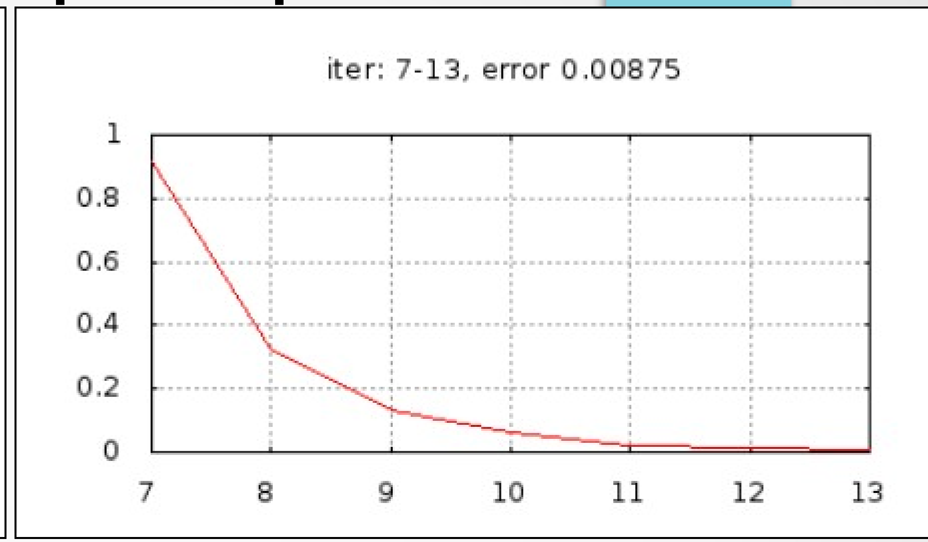


Рис.:история изменения ошибки ч.2

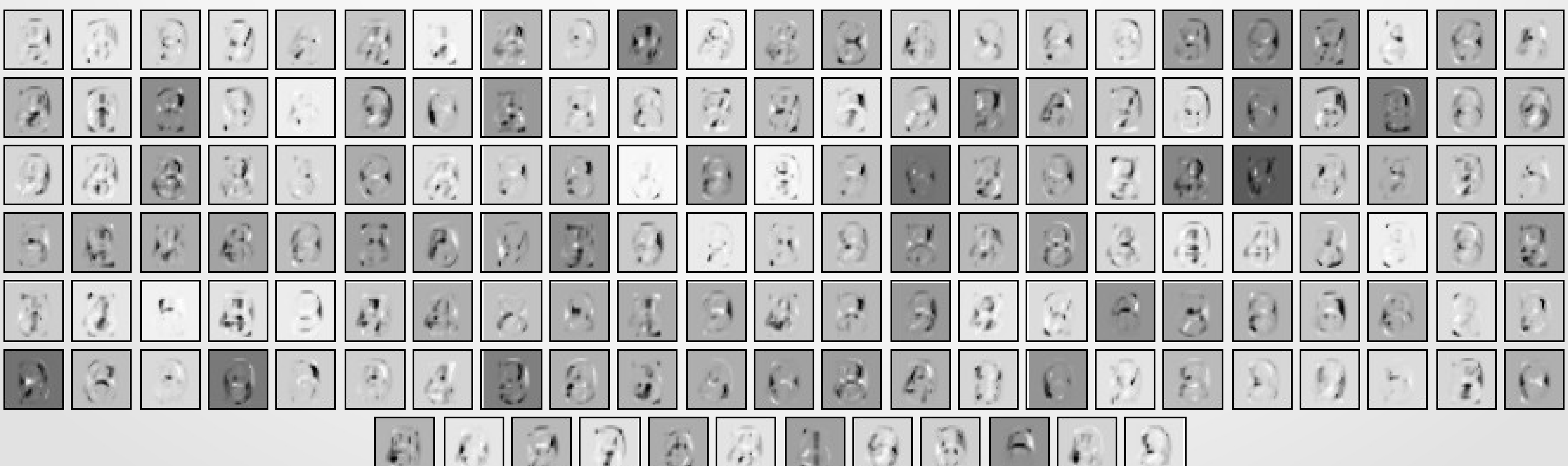


Рис.: состояния весов первого слоя

Нейросети

модификации градиентного спуска **rProp**

моменты и регуляризация не используются,
применяется простая стратегия full-batch.
параметр скорости обучения η, рассчитывается для каждого веса индивидуально

$$\eta(t) = \begin{cases} min(\eta_{max}, a \cdot \eta(t-1)) & : \quad S > 0 \\ max(\eta_{min}, b \cdot \eta(t-1)) & : \quad S < 0 \\ \eta(t-1) & : \quad S = 0 \end{cases}$$

где $S = \nabla E(t-1) \cdot \nabla E(t)$ - произведения значений градиента на этом и предыдущем шаге,
$\eta_{max} = 50$, $\eta_{min} = 10^{-6}$, $a = 1.2$, $b = 0.5$ - константы

Изменение параметров выглядит следующим образом.

$$\Delta W_t := \eta \cdot sign(\nabla E)$$

Нейросети

rProp

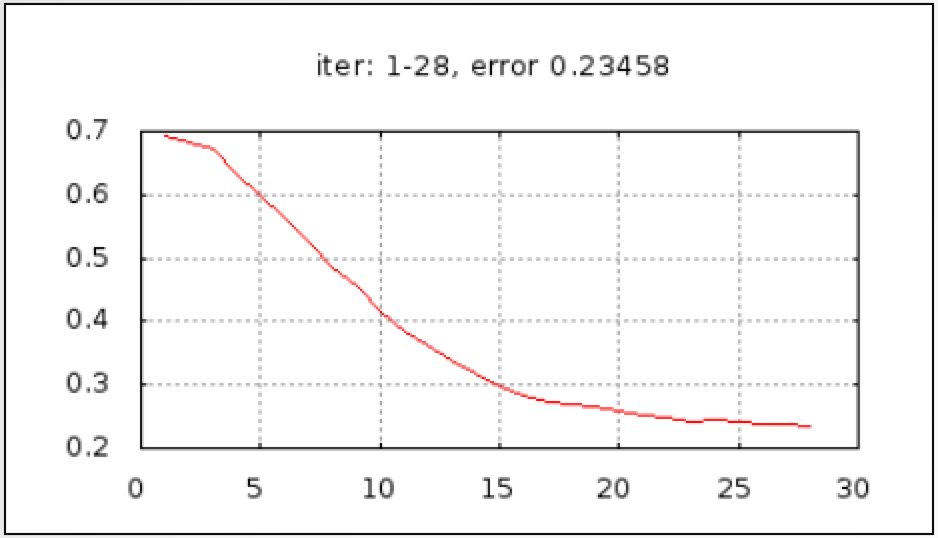


Рис.: история изменения ошибки ч.1

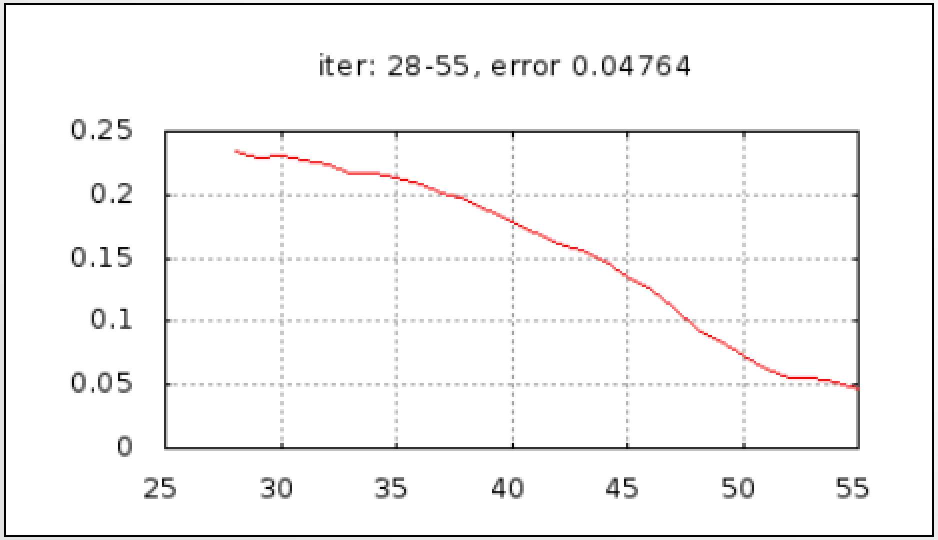


Рис.:история изменения ошибки ч.2

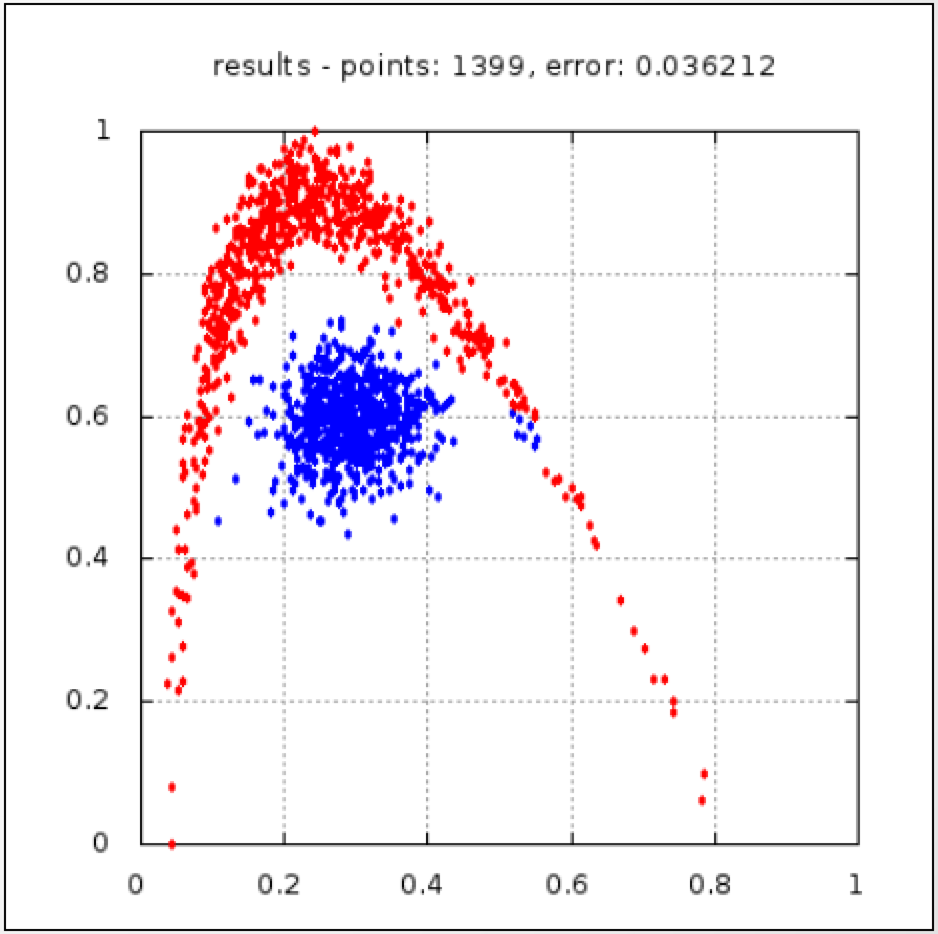


Рис.: результат теста

Нейросети

rProp

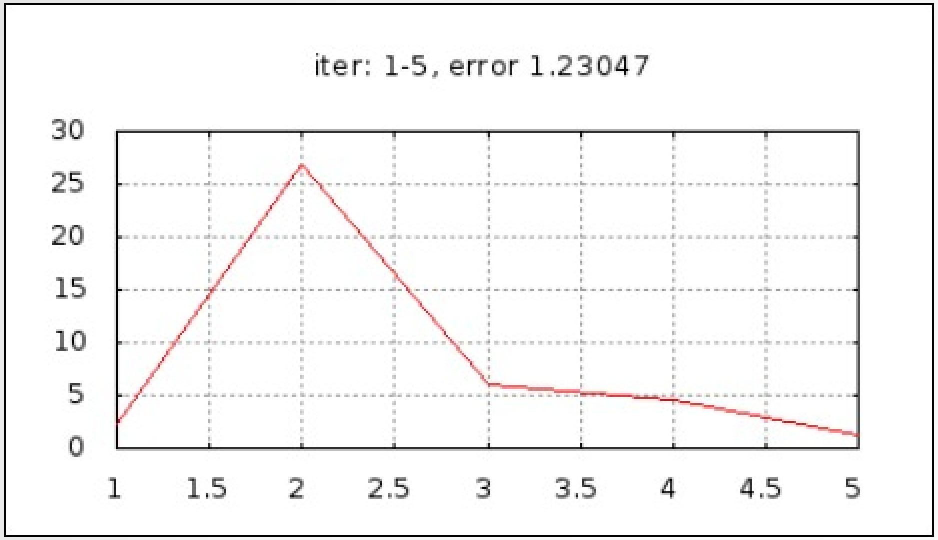


Рис.: история изменения ошибки ч.1

Рис.:история изменения ошибки ч.2

Рис.: состояния весов первого слоя

Нейросети

модификации градиентного спуска
сопряжённые градиенты (conjugate gradient)

изменения параметров выбирается таким образом,
что бы было ортогональным к предыдущему направлению

$$\Delta W := \eta \cdot (p + \rho \cdot W) + \mu \cdot \Delta W$$

коэффициент скорости обучения η,
выбирается на каждой итерации,
путём решения задачи оптимизации

$$\min_{\eta} E(\Delta W(\eta))$$

$p_0 := \nabla E$ начальное направление

$p = \nabla E + \beta \cdot p$ последующие направления изменения параметров

компенсация погрешности вычислений - сброс сопряженного направления каждые n циклов (β:=0, p:=∇E)

вычисление коэффициента сопряжения β
два основных способа

формула Флетчера-Ривса

$$\beta = \frac{g_t^T \cdot g_t}{g_{t-1}^T \cdot g_{t-1}}$$

формула Полака-Рибьера

$$\beta = \frac{g_t^T \cdot (g_t - g_{t-1})}{g_{t-1}^T \cdot g_{t-1}}$$

g:=∇E

Нейросети

сопряжённые градиенты (conjugate gradient)

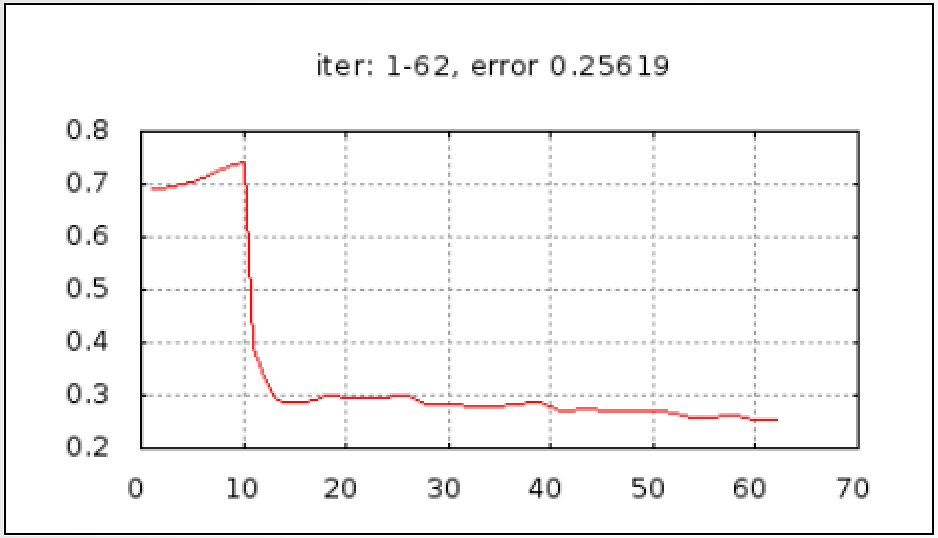


Рис.: история изменения ошибки ч.1

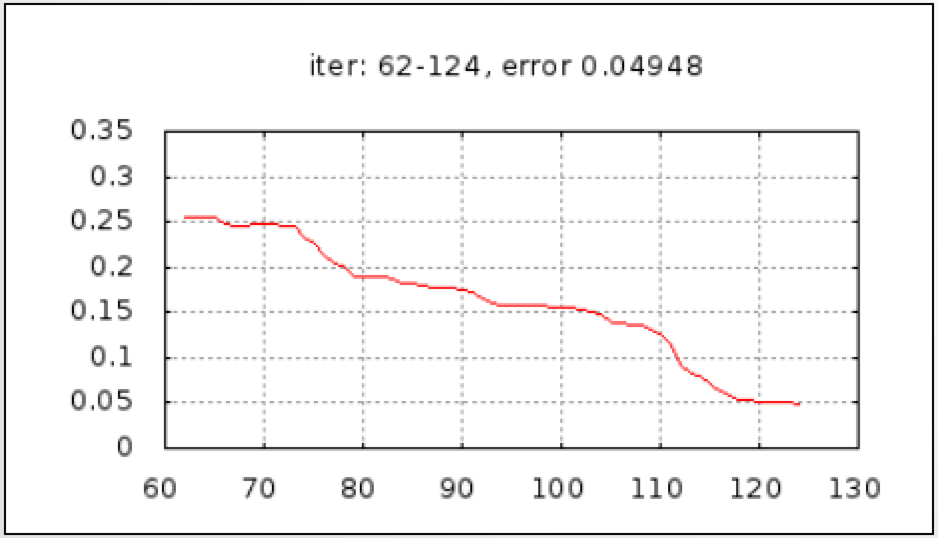


Рис.:история изменения ошибки ч.2

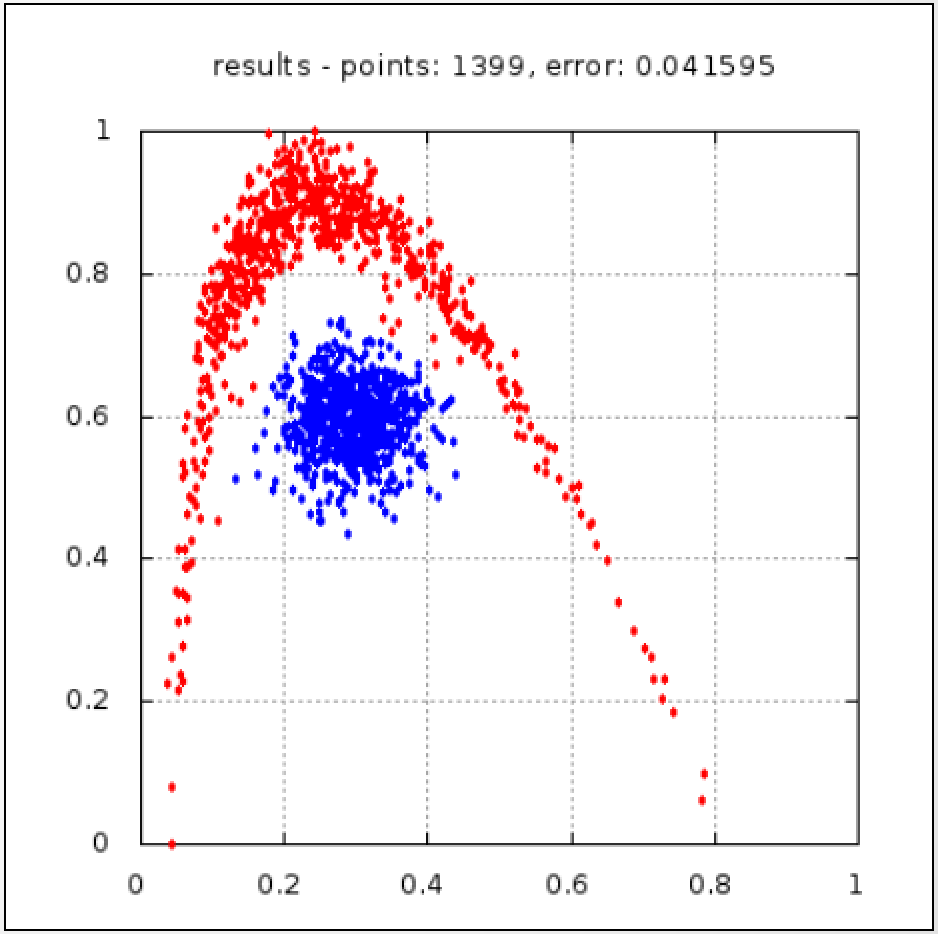


Рис.: результат теста

Нейросети

сопряжённые градиенты (conjugate gradient)

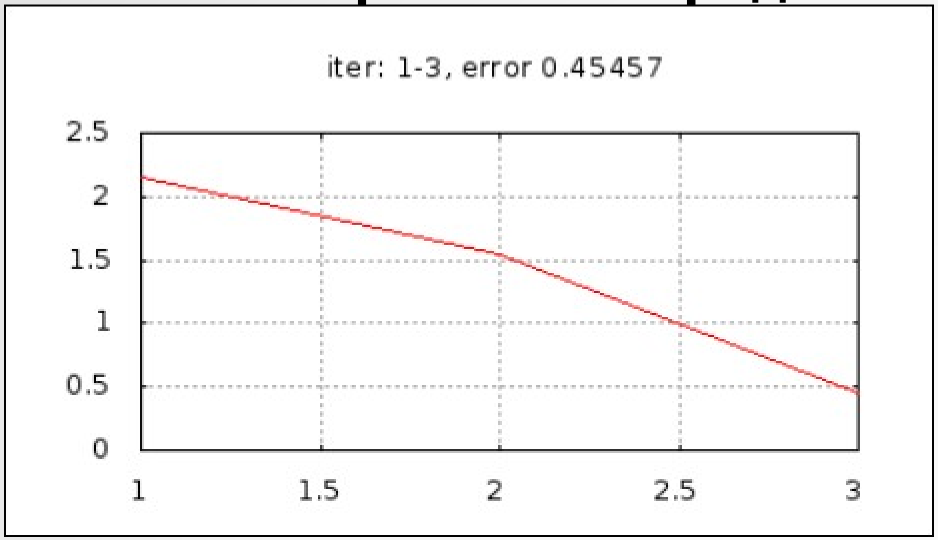


Рис.: история изменения ошибки ч.1

Рис.:история изменения ошибки ч.2

Рис.: состояния весов первого слоя

Нейросети

модификации градиентного спуска
NAG (Nesterov’s Accelerated Gradient)

градиент вычисляется относительно сдвинутых на значение момента весов

$$\Delta W_t := \eta \cdot (\nabla E(W_{t-1} + \mu \cdot \Delta W_{t-1}) + \rho \cdot W_{t-1}) + \mu \cdot \Delta W_{t-1}$$

Нейросети

модификации градиентного спуска
AdaGrad (Adaptive Gradient)

учитывает историю значений градиента следующим образом

$$g_t := \frac{\nabla E_t}{\sqrt{\sum_{i=1}^{t} \nabla E_i^2}}$$

$$\Delta W_t := \eta \cdot (g_t + \rho \cdot W_{t-1}) + \mu \cdot \Delta W_{t-1}$$

Нейросети

модификации градиентного спуска
AdaDelta

учитывает историю значений градиента и историю изменения весов следующим образом

$$S_t := \alpha \cdot S_{t-1} + (1-\alpha) \cdot \nabla E_t^2 \ ; \ S_0 := 0$$

$$D_t := \beta \cdot D_{t-1} + (1-\beta) \cdot \Delta W_{t-1}^2 \ ; \ D_0 := 0$$

$$g_t := \frac{\sqrt{D_t}}{\sqrt{S_t}} \cdot \nabla E_t$$

$$\Delta W_t := \eta \cdot (g_t + \rho \cdot W_{t-1}) + \mu \cdot \Delta W_{t-1}$$

Нейросети

градиентные методы оптимизации второго порядка

кроме градиента - направления найскорейшего роста функции, использую информацию о её кривизне

Нейросети

градиентные методы оптимизации второго порядка

кроме градиента - направления найскорейшего роста функции, использую информацию о её кривизне

$$W := W - \Delta W \qquad \Delta W = H^{-1} \cdot \nabla E$$

H - гессиан, матрица вторых производных целевой функции E

вектор градиента

$$g = \nabla E = \begin{bmatrix} \frac{\partial E}{\partial w_1} \\ \vdots \\ \frac{\partial E}{\partial w_n} \end{bmatrix}$$

$$H = \begin{bmatrix} \frac{\partial^2 E}{\partial w_1 \partial w_1} & \cdots & \frac{\partial^2 E}{\partial w_1 \partial w_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 E}{\partial w_n \partial w_1} & \cdots & \frac{\partial^2 E}{\partial w_n \partial w_n} \end{bmatrix}$$

Нейросети

градиентные методы оптимизации второго порядка

кроме градиента - направления найскорейшего роста функции, использую информацию о её кривизне

$$W := W - \Delta W \qquad \Delta W = H^{-1} \cdot \nabla E$$

H - гессиан, матрица вторых производных целевой функции E

вектор градиента

$$g = \nabla E = \begin{bmatrix} \frac{\partial E}{\partial w_1} \\ \vdots \\ \frac{\partial E}{\partial w_n} \end{bmatrix}$$

$$H = \begin{bmatrix} \frac{\partial^2 E}{\partial w_1 \partial w_1} & \cdots & \frac{\partial^2 E}{\partial w_1 \partial w_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 E}{\partial w_n \partial w_1} & \cdots & \frac{\partial^2 E}{\partial w_n \partial w_n} \end{bmatrix}$$

вычисление гессиана H это затратная процедура
можно обойтись приближением H

Нейросети

метод BFGS
или алгоритм Бройдена-Флетчера-Гольдфарба-Шанно
(Broyden–Fletcher–Goldfarb–Shanno)

$$W := W - \Delta W \qquad \Delta W = H^{-1} \cdot \nabla E$$

для вычисления обратного гессиана H^{-1}
использует изменение значений градиента ∇E и изменения весов ΔW.

вектор градиента ∇E вычисляется с помощью процедуры обратного распространения ошибки

Е.С.Борисов О методах обучения многослойных нейронных сетей прямого распространения. (Часть 3)
http://mechanoid.su/neural-net-backprop3.html

Нейросети

метод BFGS
или алгоритм Бройдена-Флетчера-Гольдфарба-Шанно
(Broyden–Fletcher–Goldfarb–Shanno)

$$W := W - \Delta W \qquad \Delta W = H^{-1} \cdot \nabla E$$

для вычисления обратного гессиана H^{-1}
использует изменение значений градиента ∇E и изменения весов ΔW.

вектор градиента ∇E вычисляется с помощью процедуры обратного распространения ошибки

приближение обратного гессиана $V \approx H^{-1}$
это матрица размера n×n (где n - длинна вектора градиента g)

значения V вычисляются на каждом шаге алгоритма следующим образом.

$$V_0 := 1$$

$$V_{k+1} := V_k - \frac{V_k \cdot s \cdot s^T \cdot V_k}{s^T \cdot V_k \cdot s} + \frac{r \cdot r^T}{s^T \cdot s}$$

r = ∇E(t) - ∇E(t-1) - изменение градиента

s = ΔW = W(t) - W(t-1) - изменение весов

Е.С.Борисов О методах обучения многослойных нейронных сетей прямого распространения. (Часть 3)
http://mechanoid.su/neural-net-backprop3.html

Нейросети

метод BFGS

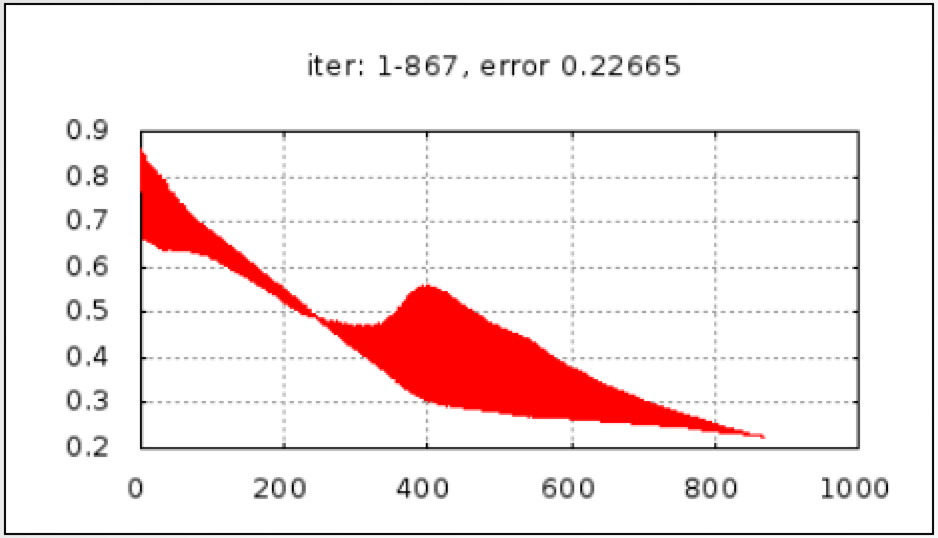


Рис.: история изменения ошибки ч.1

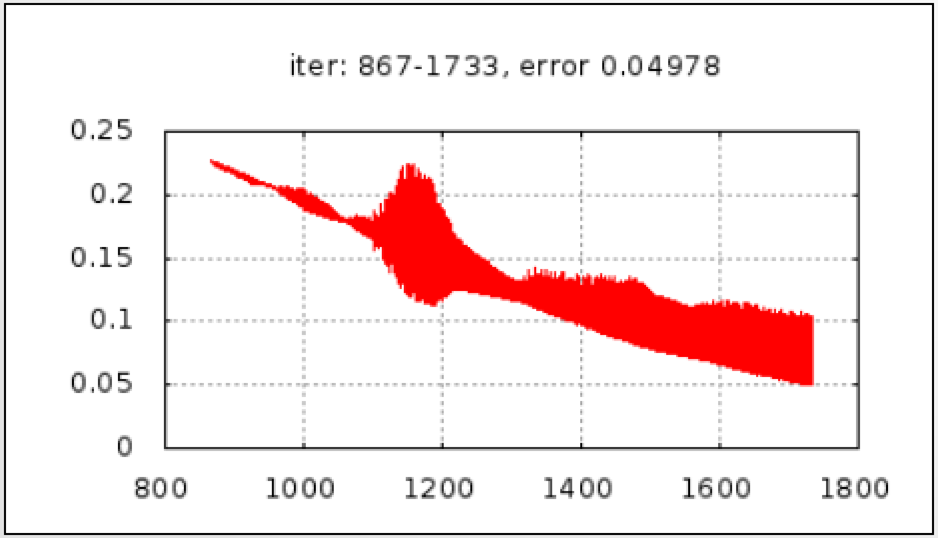


Рис.:история изменения ошибки ч.2

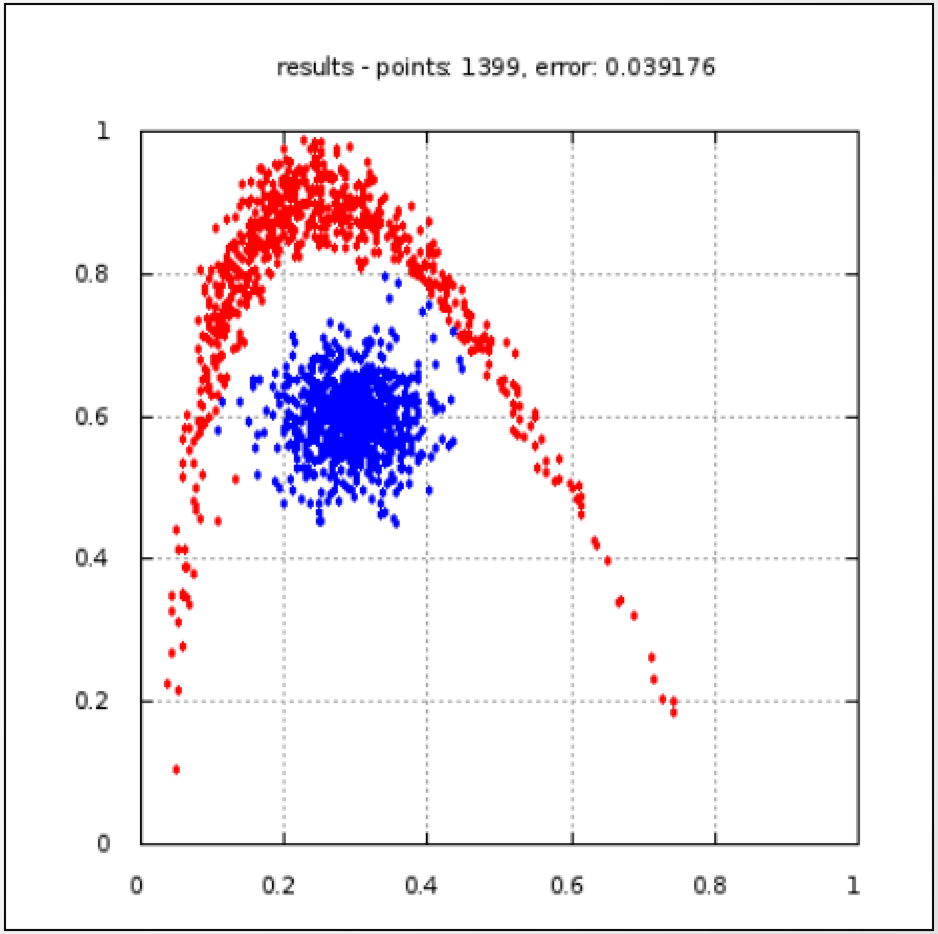


Рис.: результат теста

Нейросети

метод Левенберга-Марквардта (LMA)

вычисляем приближение гессиана H через якобиан J - матрицу первых производных,
e - ошибки сети на всей учебной выборке, M - количество выходов, P - количество примеров.

$$e = d - o = \begin{bmatrix} e_{11} \\ \vdots \\ e_{M1} \\ e_{12} \\ \vdots \\ e_{MP} \end{bmatrix}$$

$$J = \left[\frac{\partial e}{\partial w} \right]$$

Е.С.Борисов О методах обучения многослойных нейронных сетей прямого распространения. (Часть 3)
http://mechanoid.su/neural-net-backprop3.html

Нейросети

метод Левенберга-Марквардта (LMA)

вычисляем приближение гессиана H через якобиан J - матрицу первых производных,
e - ошибки сети на всей учебной выборке, M - количество выходов, P - количество примеров.

$$e = d - o = \begin{bmatrix} e_{11} \\ \vdots \\ e_{M1} \\ e_{12} \\ \vdots \\ e_{MP} \end{bmatrix}$$

$$H \approx J^T \cdot J + \mu \cdot I$$

где
J - якобиан, матрица первых производных функции ошибки,
μ - параметр,
$I = (J^T \cdot J) \circ E$ - диагональная матрица из элементов главной диагонали $(J^T \cdot J)$,
$\circ$ - поэлементное умножение матриц (Hadamard product).

Вектор градиента вычисляется следующим образом.

$$g = J^T \cdot e$$

$$J = \left[\frac{\partial e}{\partial w} \right]$$

Если собрать всё вместе то получаем следующую формулу для изменения весов сети.

$$\Delta W = (J^T \cdot J + \mu \cdot I)^{-1} \cdot J^T \cdot e$$

Е.С.Борисов О методах обучения многослойных нейронных сетей прямого распространения. (Часть 3)
http://mechanoid.su/neural-net-backprop3.html

Нейросети

метод LMA

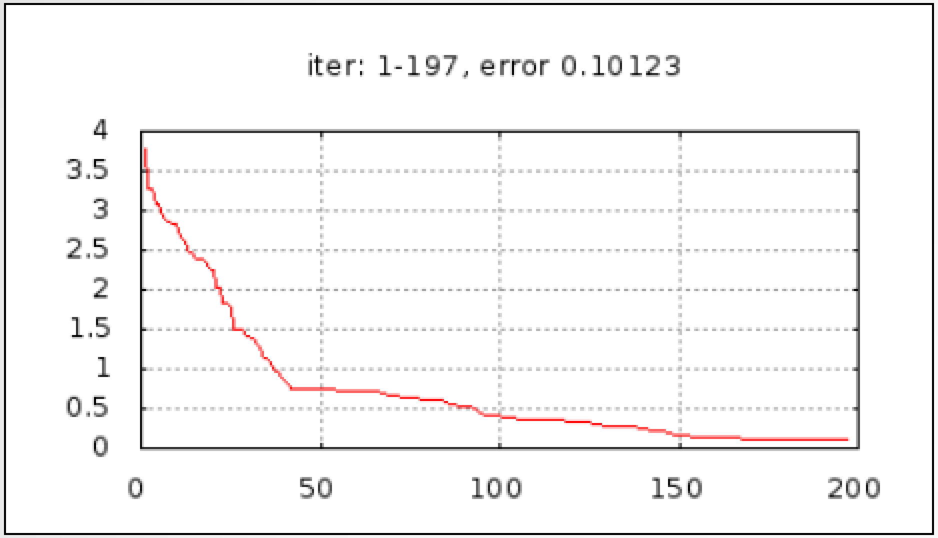


Рис.: история изменения ошибки ч.1

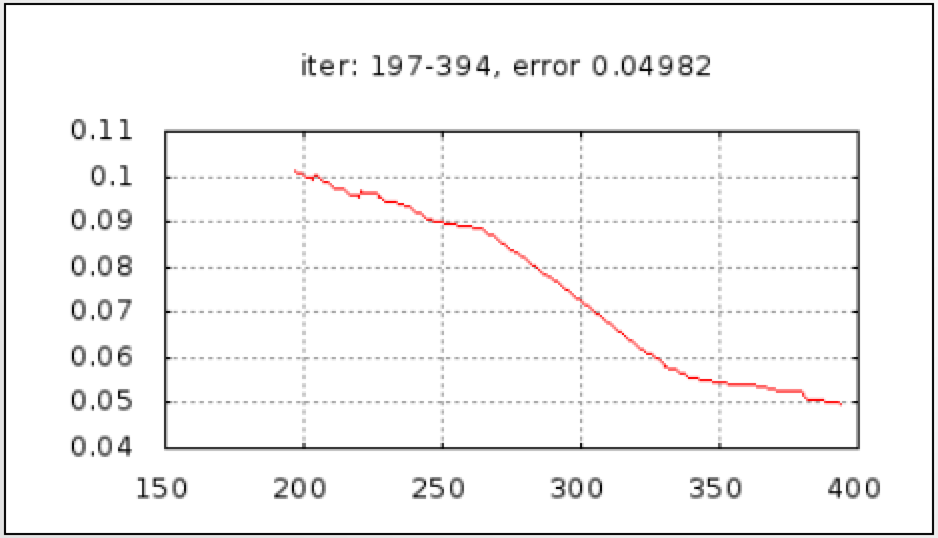


Рис.:история изменения ошибки ч.2

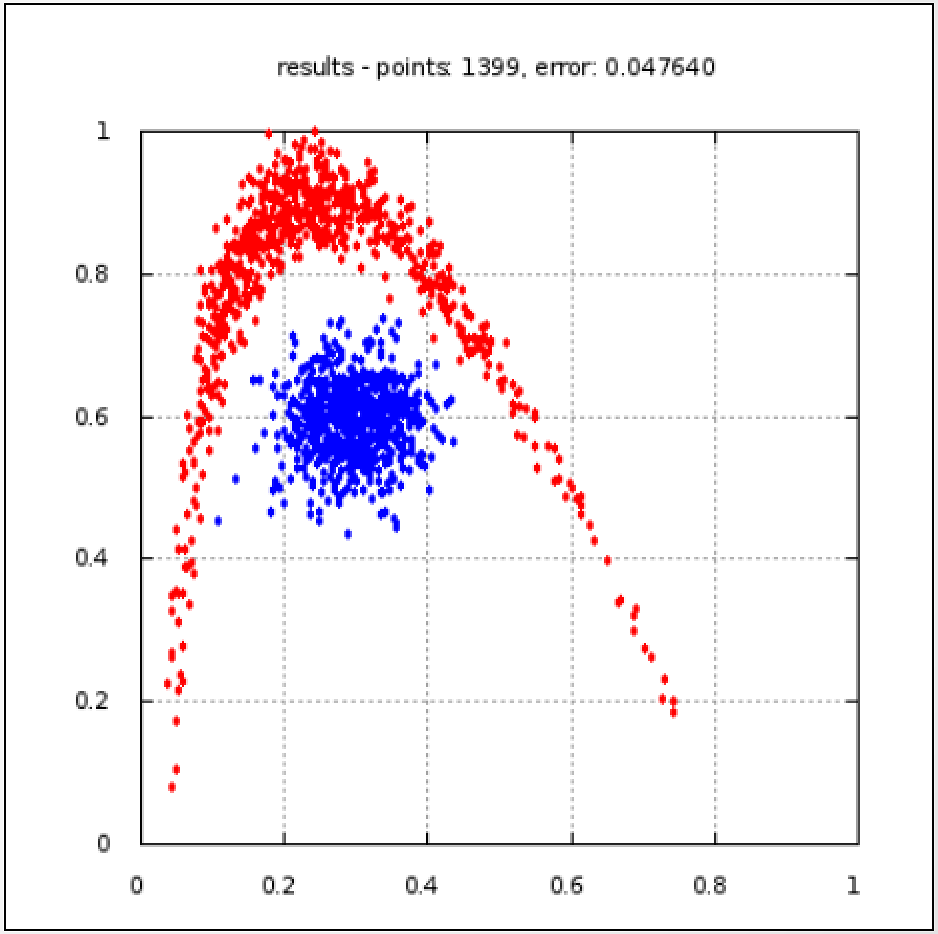


Рис.: результат теста

Нейросети

Ещё раз про количество слоёв

Если три обрабатывающих слоя строят поверхность любой формы

То зачем строить нейросети с количеством слоёв больше трёх ?

Нейросети

Ещё раз про количество слоёв

Если три обрабатывающих слоя строят поверхность любой формы

То зачем строить нейросети с количеством слоёв больше трёх ?

о концепции Deep Learning

искусственные нейронные сети с большим количеством слоёв

- много данных

- через большое количество слоёв

- с большим количеством нейронов

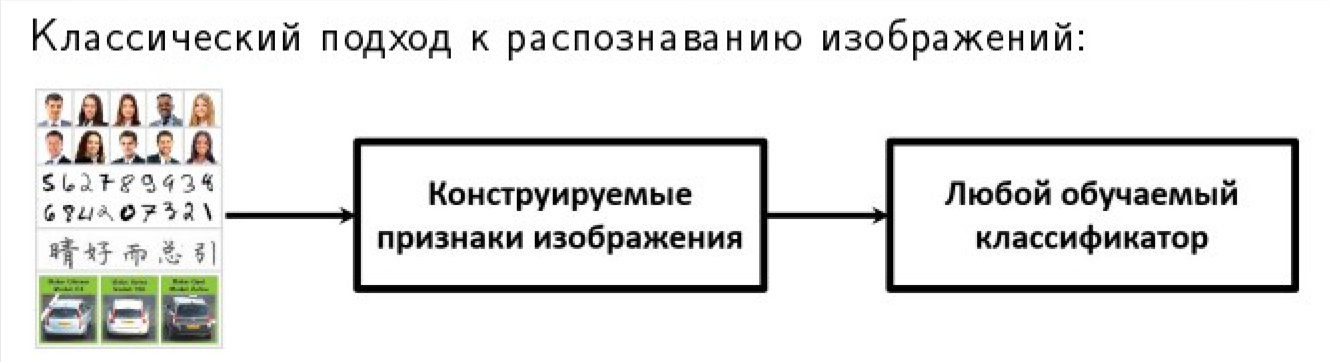
Нейросети

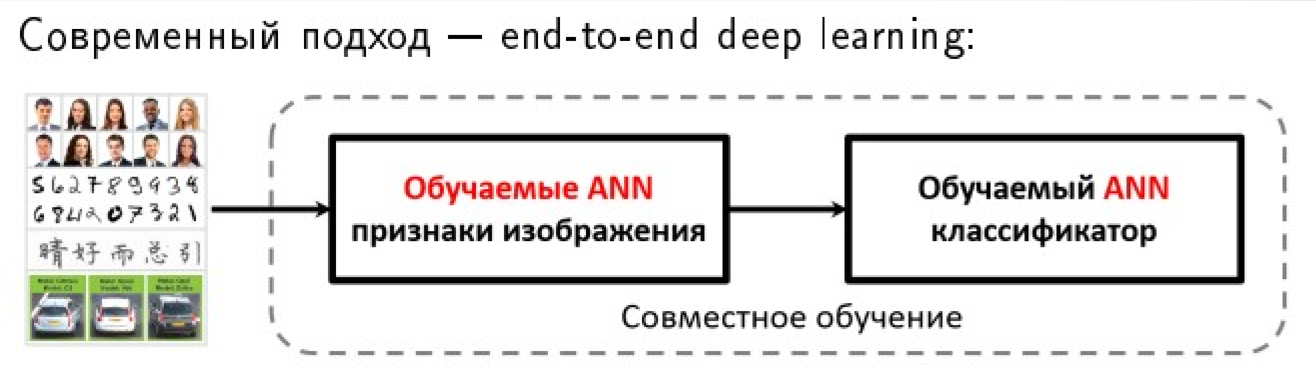
Deep Learning - многоуровневые модели интеллектуальных систем

Принцип Representation learning

Функция срытых слоёв - автоматическое извлеченеи признаков,
цель - свести к линейно разделимой задаче

последний слой — простой линейный классификатор





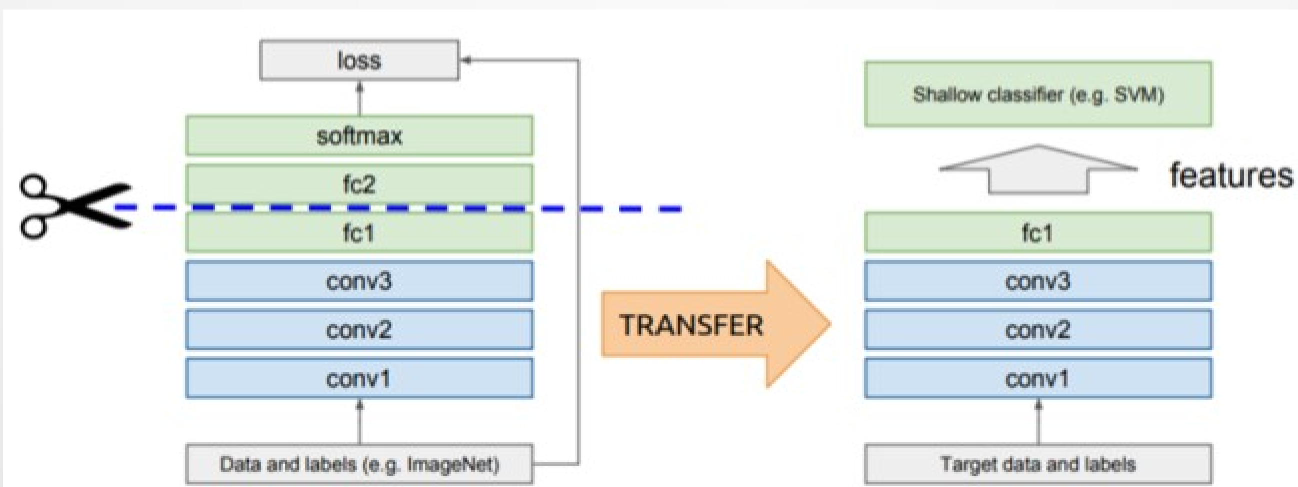
Нейросети

Transfer learning — перенос частей обученных ИНС в другие модели

Jason Yosinski, Je Clune, Yoshua Bengio, Hod Lipson. How transferable are features in deep neural networks? 2014.

Свёрточная сеть для обработки изображений:

- $z = f(x, \alpha)$ — свёрточные слои для векторизации объектов
- $y = g(z, \beta)$ — полносвязные слои под конкретную задачу



Нейросети: литература

git clone *https://github.com/mechanoid5/ml_lectorium.git*

К.В. Воронцов Нейронные сети. - курс "Машинное обучение" ШАД Яндекс 2014

Е.С.Борисов О методах обучения многослойных нейронных сетей прямого распространения. http://mechanoid.su/neural-net-backprop.html

Осовский С. Нейронные сети для обработки информации. — М.: Финансы и статистика, 2002.

Воронцов К. В.
Прикладные модели машинного обучения. 2021.
Лекция 2: Обучение без учителя.
https://www.youtube.com/watch?v=wfbe2yaXAkI