МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
**«Национальный исследовательский**
**Нижегородский государственный университет им. Н.И. Лобачевского»**
**(ННГУ)**

**Институт информационных технологий, математики и механики**

Направление подготовки: «Прикладная математика и информатика»
Магистерская программа: «Компьютерные науки и приложения»

# ОТЧЕТ

**Тема:**
**«Реализация метода обратного распространения ошибки для двухслойной полносвязной сети»**

**Выполнил(а):** студент(ка) группы 381803-4м

Синицкая Олеся Вячеславовна

Нижний Новгород
2019

## Содержание


Постановка задачи ...3

Метод обратного распространения ошибки ...4

Вывод математических формул для вычисления градиентов функции ошибки ...6

1. Вычисление производных $\frac{\partial E}{\partial \omega_{js}^{(2)}}$ ...6

   Формулы корректировки весов $\omega_{js}^{(2)}$ ...7

2. Вычисление производных $\frac{\partial E}{\partial \omega_{si}^{(1)}}$ ...8

   Формулы корректировки весов $\omega_{si}^{(1)}$ ...10

Программная реализация ...11

1. Инструкция по сборке программного кода и запуску приложения ...11
2. Описание разработанного программного кода ...11

Эксперименты ...13

# Постановка задачи

Выполнение практической работы предполагает решение ***следующих задач***:

1. Изучение общей схемы метода обратного распространения ошибки.
2. Вывод математических формул для вычисления градиентов функции ошибки по параметрам нейронной сети и формул коррекции весов.
3. Проектирование и разработка программной реализации.
4. Тестирование разработанной программной реализации.
5. Подготовка отчета, содержащего минимальный объем информации по каждому этапу выполнения работы.

# Метод обратного распространения ошибки

Метод обратного распространения ошибки определяет стратегию выбора параметров сети $w$ с использованием градиентных методов оптимизации в предположении, что целевая функция $E(w)$ непрерывна.

Градиентные методы на каждом шаге уточняют значения параметров, по которым проводится оптимизация, согласно формуле:

$$w(k+1) = w(k) + \Delta w,$$

где $\Delta w = \eta p(w)$ определяет сдвиг значений параметров, $\eta$, $0 < \eta < 1$ – ***скорость обучения*** – параметр обучения, который определяет «скорость» движения в направлении минимального значения функции, $p(w)$ – направление в многомерном пространстве параметров нейронной сети.

В классическом методе обратного распространения ошибки направление движения совпадает с направлением антиградиента $p(w) = -\nabla E(w)$.

Общая схема метода обратного распространения ошибки включает несколько основных этапов. Первоначально синаптические веса сети инициализируются определенным образом, например, нулевыми значениями или случайно из некоторого распределения.

В данной работе была выбрана инициализация весов Ксавье:

нужно умножить случайную инициализацию на:

$$\sqrt{\frac{2}{size^{l-1}}},$$

$$W = np.random.rand(size^{l}, size^{l-1}) * \sqrt{\frac{2}{size^{l-1}}}\ , l - \text{слой}, size^{l} - \text{число нейронов в слое } l$$

Далее метод работает для каждого примера обучающей выборки.

1. Прямой проход нейронной сети в направлении передачи информации от входного сигнала к скрытым слоям и выходному слою сети. На данном этапе вычисляются значения выходных сигналов нейронов скрытых слоев и выходного слоя, а также соответствующие значения производных функций активации на каждом слое сети.
2. Вычисление значения функции ошибки и градиента этой функции.
3. Обратный проход нейронной сети в направлении от выходного слоя к входному слою, и корректировка синаптических весов.
4. Повторение этапов 1 – 3 до момента выполнения критериев остановки. В качестве критериев остановки используется число итераций метода (количество проходов), либо достигнутая точность.

В процессе обучения многослойной полносвязной нейронной сети ей многократно предъявляется предопределенное множество обучающих примеров. Один полный цикл предъявления полного набора примеров называется ***эпохой***. В ходе обучения может выполняться несколько таких циклов до момента стабилизации синаптических весов, либо достижения минимального значения функции ошибки.

В данной работе используется стохастический поиск – когда изменяется порядок примеров в обучающей выборке перед каждой эпохой.

Так же используется пакетный режим. В данном режиме корректировка весов осуществляется по всем примерам эпохи. В этом случае используется функция ошибки для всего набора тренировочных данных, нормированная по числу примеров выборки. Корректировка весов также проводится по всему набору данных.

В качестве функции активации на скрытом слое используется функция LReLU.

$$\mathrm{LReLU(v)} = \begin{cases} \alpha v, v < 0, \\ v, v \geq 0 \end{cases}$$

В качестве функии активации на втором слое используется функция softmax.

$$\varphi(u_j) = \frac{e^{u_j}}{\sum_{i=1}^{M} e^{u_j}}$$

В качестве функции ошибки используется крос-энтропия.

$$E(w) = -\frac{1}{L}\sum_{k=1}^{L}\sum_{j=1}^{M} y_j^k ln u_j^k$$

```
Псевдокод:
Back_propagation(){
Инициализировать веса
    Для каждой эпохи j=1,epoch
        Для каждого батча i=1,размер входных данных
            Прямой обход (для каждого объекта из батча):
                Вычислить все v, u
                Вычислить производные фукций активаций
        Обратный проход (для всего батча)
            Вычисление целевой функции и ее градиента
            Скорректировать веса
}
```

# Вывод математических формул для вычисления градиентов функции ошибки

Вывод математических формул для вычисления градиентов функции ошибки по параметрам нейронной сети и формул коррекции весов.

## Вычисление производных $\frac{\partial \mathrm{E}}{\partial \omega_{\mathrm{js}}^{(2)}}$

$$E = -\sum_{k=1}^{M} y_k \ln u_k = -\sum_{k=1}^{M} y_k \ln \frac{e^{g_k}}{\sum_{n=1}^{M} e^{g_n}}$$

$$g_j = \sum_{s=0}^{K} \omega_{js}^{(2)} v_s$$

$$\frac{\partial E}{\partial \omega_{js}^{(2)}} = \frac{\partial E}{\partial g_j} \frac{\partial g_j}{\partial \omega_{js}^{(2)}} = \frac{\partial E}{\partial g_j} v_s$$

$$\frac{\partial E}{\partial g_j} = \left(-y_1 \ln \frac{e^{g_1}}{\sum_{n=1}^{M} e^{g_n}} - .. - \ln \frac{e^{g_j}}{\sum_{n=1}^{M} e^{g_n}} - .. - y_M \ln \frac{e^{g_M}}{\sum_{n=1}^{M} e^{g_n}}\right)'_{g_j}$$

$$\frac{\partial E}{\partial g_j} = \frac{\partial}{\partial g_j}\left(-\sum_{k=1}^{M} y_k \ln \frac{e^{g_k}}{\sum_{n=1}^{M} e^{g_n}}\right) = -\sum_{k=1}^{M} y_k \frac{\partial\left(\ln \frac{e^{g_k}}{\sum_{n=1}^{M} e^{g_n}}\right)}{\partial g_j} = \left(\text{обозначим } \frac{e^{g_k}}{\sum_{n=1}^{M} e^{g_n}} = d_k\right)$$

$$= -\sum_{k=1}^{M} y_k \frac{1}{d_k} \frac{\partial d_k}{\partial g_j} \overset{(1)}{=}$$

$$\overset{(1)}{=} - y_j \frac{1}{\varphi^{(2)}(g_j)} \varphi^{(2)}(g_j)\left(1 - \varphi^{(2)}(g_j)\right) - \sum_{k=1,k\neq j}^{M} y_k \frac{1}{\varphi^{(2)}(g_k)} \left(-\varphi^{(2)}(g_k)\varphi^{(2)}(g_j)\right) =$$

$$= -y_j + y_j\varphi^{(2)}(g_j) + \sum_{k=1,k\neq j}^{M} y_k\varphi^{(2)}(g_j) = -y_j + \left(y_j + \sum_{k=1,k\neq j}^{M} y_k\right)\varphi^{(2)}(g_j) = \boxed{-y_j + \varphi^{(2)}(g_j) = \frac{\partial E}{\partial g_j}}.$$

$$(1):\ \frac{\partial d_k}{\partial g_j} = \begin{cases} \frac{e^{g_k}}{\sum_{n=1}^{M} e^{g_n}} = \frac{e^{g_j}\sum_{n=1}^{M} e^{g_n} - e^{g_j}e^{g_j}}{\left(\sum_{n=1}^{M} e^{g_n}\right)^2} = \frac{e^{g_j}}{\sum_{n=1}^{M} e^{g_n}} - \frac{e^{g_j}e^{g_j}}{\left(\sum_{n=1}^{M} e^{g_n}\right)^2} = \\ = \varphi^{(2)}(g_j) - (\varphi^{(2)}(g_j))^2 = \varphi^{(2)}(g_j)\left(1 - \varphi^{(2)}(g_j)\right), k = j \\ -\frac{e^{g_k}e^{g_j}}{\left(\sum_{n=1}^{M} e^{g_n}\right)^2} = -\varphi^{(2)}(g_k)\varphi^{(2)}(g_j), k \neq j \end{cases}$$

*Получаем:*

$$\frac{\partial E}{\partial \omega_{js}^{(2)}} = \left(-y_j + \varphi^{(2)}(g_j)\right) v_s$$

$$\varphi^{(2)}(g_j^k) = u_j^k$$

# Формулы корректировки весов

$\omega_{js}^{(2)} = \omega_{js}^{(2)} - \frac{\eta}{L}\sum_{k=1}^{L}\left(-y_j^k + \varphi^{(2)}\left(g_j^k\right)\right)v_s^k$ или $\omega_{js}^{(2)} = \omega_{js}^{(2)} - \frac{\eta}{L}\sum_{k=1}^{L}\left(-y_j^k + u_j^k\right)v_s^k$

$\Delta\omega_{js}^{(2)} = -\frac{\eta}{L}\sum_{k=1}^{L}\left(-y_j^k + \varphi^{(2)}\left(g_j^k\right)\right)v_s^k$

$\omega_{js}^{(2)} = \omega_{js}^{(2)} + \Delta\omega_{js}^{(2)}$

*В матричном виде:*

$\Delta W^{(2)} = -\frac{\eta}{L}(U - Y)^T V$

$W^{(2)} = W^{(2)} + \Delta W^{(2)}$

## *Вычисление производных* $\frac{\partial E}{\partial \omega_{si}^{(1)}}$

$$E = -\sum_{k=1}^{M} y_k \ln u_k = -\sum_{k=1}^{M} y_k \ln \frac{e^{g_k}}{\sum_{n=1}^{M} e^{g_n}}$$

$$g_j = \sum_{s=0}^{K} \omega_{js}^{(2)} v_s = \sum_{s=0}^{K} \omega_{js}^{(2)} \varphi^{(1)}(f_s) = \sum_{s=0}^{K} \omega_{js}^{(2)} \varphi^{(1)}(\sum_{i=0}^{N} \omega_{si}^{(1)} x_i)$$

*Вычислим производные*

*Коротко*

$$\frac{\partial E}{\partial \omega_{si}^{(1)}} = \sum_{k=1}^{M} \frac{\partial E}{\partial g_j} \frac{\partial g_j}{\partial v_s} \frac{\partial v_s}{\partial f_s} \frac{\partial f_s}{\partial \omega_{si}^{(1)}} = \sum_{k=1}^{M} \frac{\partial E}{\partial g_j} \frac{\partial g_j}{\partial v_s} \frac{\partial v_s}{\partial f_s} x_i = \sum_{k=1}^{M} (-y_j + \varphi^{(2)}(g_j)) \omega_{js}^{(2)} \frac{\partial \varphi^{(1)}(f_s)}{\partial f_s} x_i .$$

$$\frac{\partial g_j}{\partial v_s} = \frac{\partial \sum_{s=0}^{K} \omega_{js}^{(2)} v_s}{\partial v_s} = \omega_{js}^{(2)}$$

$$\varphi^{(1)}(f_s) = \begin{cases} \alpha f_s, f_s < 0 \\ f_s, f_s \geq 0 \end{cases}$$

$$\frac{\partial v_s}{\partial f_s} = \frac{\partial \varphi^{(1)}(f_s)}{\partial f_s} = \begin{cases} \alpha, f_s < 0 \\ 1, f_s \geq 0 \end{cases}$$

$$\varphi^{(1)}\left(\sum_{i=0}^{N} \omega_{si}^{(1)} x_i\right) = \varphi^{(1)}(f_s) = \begin{cases} \alpha f_s, f_s < 0 \\ f_s, f_s \geq 0 \end{cases} = \begin{cases} \alpha \sum_{i=0}^{N} \omega_{si}^{(1)} x_i, \sum_{i=0}^{N} \omega_{si}^{(1)} x_i < 0 \\ \sum_{i=0}^{N} \omega_{si}^{(1)} x_i, \sum_{i=0}^{N} \omega_{si}^{(1)} x_i \geq 0 \end{cases} .$$

$x$ – входной слой

N – число нейронов на входном слое

$\omega_{si}^{(1)}$ - веса от входных нейронов к нейронам скрытого слоя

$v$ – скрытый слой

$K$ - число нейронов на скрытом слое

$\omega_{js}^{(2)}$ – веса от скрытых нейронов к нейронам выходного слоя

$\varphi^{(1)}$ – функция активации на скрытом слое

$u$ – выходной слой

$M$ - число нейронов на выходном слое

$\varphi^{(2)}$ - функция активации на выходном слое

$$f_s = \sum_{i=0}^{N} \omega_{si}^{(1)} x_i$$

*Подробнее*

$$\frac{\partial E}{\partial \omega_{si}^{(1)}} = \frac{\partial(-\sum_{k=1}^{M} y_k \ln u_k)}{\partial \omega_{si}^{(1)}} = -\sum_{k=1}^{M} y_k \frac{\partial(\ln u_k)}{\partial \omega_{si}^{(1)}} = -\sum_{k=1}^{M} \frac{y_k}{u_k} \frac{\partial(\frac{e^{g_k}}{\sum_{n=1}^{M} e^{g_n}})}{\partial \omega_{si}^{(1)}} =$$

$$\frac{e^{g_k}}{\sum_{n=1}^{M} e^{g_n}} = \frac{e^{\sum_{s=0}^{K} \omega_{ks}^{(2)} \varphi^{(1)}\left(\sum_{i=0}^{N} \omega_{si}^{(1)} x_i\right)}}{\sum_{n=1}^{M} e^{g_n}} = \frac{a}{b}$$

$$\frac{\partial \frac{a}{b}}{\partial \omega_{si}^{(1)}} = \frac{\frac{\partial a}{\partial \omega_{si}^{(1)}} b - a \frac{\partial b}{\partial \omega_{si}^{(1)}}}{b^2}$$

$$\frac{\partial a}{\partial \omega_{si}^{(1)}} = \frac{\partial e^{g_k}}{\partial g_k} = e^{g_k} \frac{\partial g_k}{\partial v_s} = e^{g_k} \frac{\partial \sum_{s=0}^{K} \omega_{ks}^{(2)} \varphi^{(1)} \left(\sum_{i=0}^{N} \omega_{si}^{(1)} x_i\right)}{\partial v_s} = e^{g_k} \omega_{ks}^{(2)} \frac{\partial \varphi^{(1)}(f_s)}{\partial f_s} =$$

$$= e^{g_k} \omega_{ks}^{(2)} \begin{cases} \alpha, f_s < 0 \\ 1, f_s \geq 0 \end{cases} \frac{\partial f_s}{\partial \omega_{si}^{(1)}} = e^{g_k} \omega_{ks}^{(2)} \begin{cases} \alpha, f_s < 0 \\ 1, f_s \geq 0 \end{cases} x_i$$

$$\frac{\partial b}{\partial \omega_{si}^{(1)}} = \frac{\partial \sum_{n=1}^{M} e^{g_n}}{\partial g_n} = \sum_{n=1}^{M} \frac{\partial e^{g_n}}{\partial g_n} = \sum_{n=1}^{M} e^{g_n} \frac{\partial g_n}{\partial v_s} = \sum_{n=1}^{M} e^{g_n} \frac{\partial \sum_{s=0}^{K} \omega_{ns}^{(2)} \varphi^{(1)} \left(\sum_{i=0}^{N} \omega_{si}^{(1)} x_i\right)}{\partial v_s}$$

$$= \sum_{n=1}^{M} e^{g_n} \omega_{ns}^{(2)} \frac{\partial \varphi^{(1)}(f_s)}{\partial f_s} = \sum_{n=1}^{M} e^{g_n} \omega_{ns}^{(2)} \begin{cases} \alpha, f_s < 0 \\ 1, f_s \geq 0 \end{cases} \frac{\partial f_s}{\partial \omega_{si}^{(1)}}$$

$$= \sum_{n=1}^{M} e^{g_n} \omega_{ns}^{(2)} \begin{cases} \alpha, f_s < 0 \\ 1, f_s \geq 0 \end{cases} \frac{\partial \sum_{i=0}^{N} \omega_{si}^{(1)} x_i}{\partial \omega_{si}^{(1)}} = \sum_{n=1}^{M} e^{g_n} \omega_{ns}^{(2)} \begin{cases} \alpha, f_s < 0 \\ 1, f_s \geq 0 \end{cases} x_i.$$

$$\frac{\partial \frac{a}{b}}{\partial \omega_{si}^{(1)}} = \frac{\frac{\partial a}{\partial \omega_{si}^{(1)}} b - a \frac{\partial b}{\partial \omega_{si}^{(1)}}}{b^2} = \frac{e^{g_k} \omega_{ks}^{(2)} \begin{cases} \alpha, f_s < 0 \\ 1, f_s \geq 0 \end{cases} x_i \sum_{n=1}^{M} e^{g_n} - e^{g_k} \sum_{n=1}^{M} e^{g_n} \omega_{ns}^{(2)} \begin{cases} \alpha, f_s < 0 \\ 1, f_s \geq 0 \end{cases} x_i}{\left(\sum_{n=1}^{M} e^{g_n}\right)^2} =$$

$$= \varphi^{(2)}(g_k) \frac{\omega_{ks}^{(2)} \begin{cases} \alpha, f_s < 0 \\ 1, f_s \geq 0 \end{cases} x_i \sum_{n=1}^{M} e^{g_n}}{\left(\sum_{n=1}^{M} e^{g_n}\right)} - \frac{e^{g_k} \sum_{n=1}^{M} e^{g_n} \omega_{ns}^{(2)} \begin{cases} \alpha, f_s < 0 \\ 1, f_s \geq 0 \end{cases} x_i}{\left(\sum_{n=1}^{M} e^{g_n}\right)^2} =$$

$$= \varphi^{(2)}(g_k) \omega_{ks}^{(2)} \begin{cases} \alpha, f_s < 0 \\ 1, f_s \geq 0 \end{cases} x_i - \frac{e^{g_k} \sum_{n=1}^{M} e^{g_n} \omega_{ns}^{(2)} \begin{cases} \alpha, f_s < 0 \\ 1, f_s \geq 0 \end{cases} x_i}{\left(\sum_{n=1}^{M} e^{g_n}\right)^2} = \frac{\partial \frac{e^{g_k}}{\sum_{n=1}^{M} e^{g_n}}}{\partial \omega_{si}^{(1)}} \quad (1)$$

*Напоминаем, что* $\varphi^{(1)}$ *выглядит так:* $\varphi^{(1)} \left(\sum_{i=0}^{N} \omega_{si}^{(1)} x_i\right) = \varphi^{(1)}(f_s) = \begin{cases} \alpha f_s, f_s < 0 \\ f_s, f_s \geq 0 \end{cases} =$

$$\begin{cases} \alpha \sum_{i=0}^{N} \omega_{si}^{(1)} x_i, \sum_{i=0}^{N} \omega_{si}^{(1)} x_i < 0 \\ \sum_{i=0}^{N} \omega_{si}^{(1)} x_i, \sum_{i=0}^{N} \omega_{si}^{(1)} x_i \geq 0 \end{cases}.$$

*Собираем полученные результаты (1):*

$$\frac{\partial E}{\partial \omega_{si}^{(1)}} = \frac{\partial \left(-\sum_{k=1}^{M} y_k \ln u_k\right)}{\partial \omega_{si}^{(1)}} = -\sum_{k=1}^{M} y_k \frac{\partial (\ln u_k)}{\partial \omega_{si}^{(1)}} = -\sum_{k=1}^{M} \frac{y_k}{u_k} \frac{\partial \left(\frac{e^{g_k}}{\sum_{n=1}^{M} e^{g_n}}\right)}{\partial \omega_{si}^{(1)}} =$$

$$= -\sum_{k=1}^{M} \frac{y_k}{u_k} \left( \varphi^{(2)}(g_k) \omega_{ks}^{(2)} \begin{cases} \alpha, f_s < 0 \\ 1, f_s \geq 0 \end{cases} x_i - \frac{e^{g_k} \sum_{n=1}^{M} e^{g_n} \omega_{ns}^{(2)} \begin{cases} \alpha, f_s < 0 \\ 1, f_s \geq 0 \end{cases} x_i}{\left(\sum_{n=1}^{M} e^{g_n}\right)^2} \right) =$$

$$= -\sum_{k=1}^{M} \left( \frac{y_k}{u_k} \varphi^{(2)}(g_k) \omega_{ks}^{(2)} \begin{cases} \alpha, f_s < 0 \\ 1, f_s \geq 0 \end{cases} x_i - \frac{y_k}{u_k} \frac{e^{g_k} \sum_{n=1}^{M} e^{g_n} \omega_{ns}^{(2)} \begin{cases} \alpha, f_s < 0 \\ 1, f_s \geq 0 \end{cases} x_i}{\left(\sum_{n=1}^{M} e^{g_n}\right)^2} \right) =$$

$$= -\begin{cases} \alpha, f_s < 0 \\ 1, f_s \geq 0 \end{cases} x_i \sum_{k=1}^{M} \left( y_k \omega_{ks}^{(2)} - \frac{y_k}{u_k} \frac{e^{g_k} \sum_{n=1}^{M} e^{g_n} \omega_{ns}^{(2)}}{\left(\sum_{n=1}^{M} e^{g_n}\right)^2} \right)$$

$$= -\begin{cases} \alpha, f_s < 0 \\ 1, f_s \geq 0 \end{cases} x_i \sum_{k=1}^{M} \left( y_k \omega_{ks}^{(2)} - \frac{y_k}{u_k} \varphi^{(2)}(g_k) \frac{\sum_{n=1}^{M} e^{g_n} \omega_{ns}^{(2)}}{\sum_{n=1}^{M} e^{g_n}} \right) =$$

$$= -\begin{cases}\alpha, f_s < 0\\ 1, f_s \geq 0\end{cases} x_i \sum_{k=1}^{M}(y_k\,\omega_{ks}^{(2)} - y_k \frac{\sum_{n=1}^{M} e^{g_n}\,\omega_{ns}^{(2)}}{\sum_{n=1}^{M} e^{g_n}}) = -\begin{cases}\alpha, f_s < 0\\ 1, f_s \geq 0\end{cases} x_i \left(\sum_{k=1}^{M} y_k\,\omega_{ks}^{(2)} - \sum_{k=1}^{M}(y_k \frac{\sum_{n=1}^{M} e^{g_n}\,\omega_{ns}^{(2)}}{\sum_{n=1}^{M} e^{g_n}})\right) =$$

$$= -\begin{cases}\alpha, f_s < 0\\ 1, f_s \geq 0\end{cases} x_i \left(\sum_{k=1}^{M} y_k\,\omega_{ks}^{(2)} - 1 * \frac{\sum_{n=1}^{M} e^{g_n}\omega_{ns}^{(2)}}{\sum_{n=1}^{M} e^{g_n}}\right) = -\begin{cases}\alpha, f_s < 0\\ 1, f_s \geq 0\end{cases} x_i \left(\sum_{k=1}^{M} y_k\,\omega_{ks}^{(2)} - \frac{\sum_{k=1}^{M} e^{g_k}\omega_{ks}^{(2)}}{\sum_{n=1}^{M} e^{g_n}}\right) =$$

$$-\begin{cases}\alpha, f_s < 0\\ 1, f_s \geq 0\end{cases} x_i \left(\sum_{k=1}^{M} \left(y_k\omega_{ks}^{(2)} - \frac{e^{g_k}\omega_{ks}^{(2)}}{\sum_{n=1}^{M} e^{g_n}}\right)\right) = -\begin{cases}\alpha, f_s < 0\\ 1, f_s \geq 0\end{cases} x_i \left(\sum_{k=1}^{M} \omega_{ks}^{(2)} \left(y_k - \frac{e^{g_k}}{\sum_{n=1}^{M} e^{g_n}}\right)\right) =$$

$$-\begin{cases}\alpha, f_s < 0\\ 1, f_s \geq 0\end{cases} x_i \left(\sum_{k=1}^{M} \omega_{ks}^{(2)} \left(y_k - \varphi^{(2)}(g_k)\right)\right) = \begin{cases}\alpha, f_s < 0\\ 1, f_s \geq 0\end{cases} x_i \left(\sum_{k=1}^{M} \omega_{ks}^{(2)} \left(\varphi^{(2)}(g_k) - y_k\right)\right) = \frac{\partial E}{\partial \omega_{si}^{(1)}}.$$

## Формулы корректировки весов

$$\omega_{si}^{(1)} = \omega_{si}^{(1)} - \frac{\eta}{L} \sum_{k=1}^{L} \frac{\partial \varphi^{(1)}(f_s)}{\partial f_s}\bigg|_{x^k} x_i^k \left(\sum_{j=1}^{M} \omega_{js}^{(2)} \left(\varphi^{(2)}(g_j^k) - y_j^k\right)\right), \qquad \frac{\partial \varphi^{(1)}(f_s)}{\partial f_s} = \begin{cases}\alpha, f_s < 0\\ 1, f_s \geq 0\end{cases}$$

$$\varphi^{(1)}(g_j^k) = v_j^k$$

$$\Delta\omega_{si}^{(1)} = -\frac{\eta}{L} \sum_{k=1}^{L} \frac{\partial \varphi^{(1)}(f_s)}{\partial f_s}\bigg|_{x^k} x_i^k \left(\sum_{j=1}^{M} \omega_{js}^{(2)} \left(\varphi^{(2)}(g_j^k) - y_j^k\right)\right)$$

$$\omega_{si}^{(1)} = \omega_{si}^{(1)} + \Delta\omega_{si}^{(1)}$$

*В матричном виде:*

$\Delta W^{(2)} = -\frac{\eta}{L}([(U - Y) \cdot W^{(2)}] * \Phi)^T \cdot X$, где $*$ — поэлементное умножение.

$W^{(2)} = W^{(2)} + \Delta W^{(2)}$

## Программная реализация

## Инструкция по сборке программного кода и запуску приложения на данных базы MNIST

Перед запуском необходимо установить пакеты tensorflow, keras, numpy, sklearn.
Помощь по установке этих и других пакетов для Windows вы можете найти здесь:
https://github.com/jeffheaton/t81_558_deep_learning/blob/master/t81_558_class_01_1_overview.ipynb.
Для запуска приложения можно воспользоваться Jupyter Notebook (файл backpropagation.ipynb).
Для запуска с Anaconda Prompt нужно создать окружение(enviroment), перейти в него и запустить backpropagation.py.

## Описание разработанного программного кода

Рассмотрим класс network. Для того, чтобы обучить сеть, необходимо создать экземпляр класса network. Экземпляр класса содержит:

- self.speed – скорость обучения
- self.batch - батч

#число нейронов входного слоя

- self.num_input_neurons_x

#число нейронов скрытого слоя

- self.num_hidden_neurons_v = num_hidden_neurons_v
- self.V – нейроны скрытого слоя

#число нейронов выходного слоя

- self.num_output_neurons_u = num_output_neurons_u
- self.U – нейроны выходного слоя

- self.w2 – веса от скрытого слоя к выходному
- self.w1 – веса от входного слоя к скрытому

- self.der_LReLU – производные от функции LReLU скрытого слоя

Функция *run* – пример использования класса network.
Функция *run* создает экземпляр класса network и вызывает функцию обучения *fit*.
Функция *run(num_hidden_neurons_v, epoch, batch, speed_train* ) показывает как можно обучить нейронную сеть, используя данные MNIST и проверить результат на тренировочной и тестовой выборках. Функция *run* принимает на вход:

- num_hidden_neurons_v – число нейронов на скрытом слое
- epoch – чило эпох
- batch – размер батча
- speed_train – скорость обучения

Функция *fit* обеспечивает обучение и тестирование сети, получая на вход выборки, размер пачек, скорость обучения и количество эпох.

*fit* реализует стохастический пакетный режим обучения и выбор весов с помощью метода обратного распространения ошибки.
Функция *fit* принимает на вход:

- x_train – входные данные
- y_train - правильные ответы(метки)
- batch – размер пачки
- speed_train – скорость обучения
- epoch – количество эпох

# Эксперименты

| Скорость обучения | батч | Количество эпох | Число скрытых нейронов | Время | Тренировочная выборка Точность | Тренировочная выборка Потери | Тестовая выборка Точность | Тестовая выборка Потери |
|---|---|---|---|---|---|---|---|---|
| 0.1 | 10 | 10 | 80 | 01:33.460671 | 99.045 | 0.029118291203606057 | 97.57000000000001 | 0.08896567283046788 |
| 0.1 | 10 | 20 | 80 | 03:05.735970 | 99.905 | 0.004198767634376354 | 97.96000000000001 | 0.09408642342944606 |
| 0.1 | 120 | 20 | 80 | 01:45.403496 | 98.31833333333333 | 0.0599678358397 | 97.19 | 0.0947928902596885 |
| 0.1 | 32 | 20 | 300 | 08:30.556453 | 99.955 | 0.006391172703401912 | 98.22999999999999 | 0.05951493573074625 |
| 0.1 | 128 | 20 | 300 | 06:23.963461 | 98.86 | 0.04522896811537448 | 97.83 | 0.07237803520961507 |