|  |
| --- |
| МИНИСТЕРСТВО НАУКИ И ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  федеральное государственное АВТОНОМНОЕ образовательное учреждение высшего образования  «Национальный исследовательский ядерный университет «МИФИ» |
| **Технологический институт –**  филиал федерального государственного автономного образовательного учреждения высшего образования «Национальный исследовательский ядерный университет «МИФИ»  **(ТИ НИЯУ МИФИ)** |

Кафедра ТЕХНИЧЕСКИХ СИСТЕМ КОНТРОЛЯ И УПРАВЛЕНИЯ

Отчет по лабораторной работе №1

**Создание персептронной нейронной сети применимо к задаче классификации символьных элементов**

Вариант № 3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Проверил |  |  |  |  |
| ст.преподаватель |  |  |  | А.К. Кревский |
| *(должность)* |  | *(подпись)* |  | *(И.О. Фамилия)* |
| Студент |  |  |  |  |
|  |  |  |  |  |
| *(группа)* |  | *(подпись)* |  | *(И.О. Фамилия)* |

Работа защищена с оценкой «\_\_\_\_\_\_\_\_\_\_\_» «\_\_\_» «\_\_\_» \_\_\_\_\_\_\_\_20 г.

Лесной

2018

Цель работы: Изучить работу перцептронной нейронной сети, ознакомиться с правилом Хоба, добиться работы собственной перцептронной нейронной сети для классификации символьных элементов

Выполняемый код:

**import random**

**#Numbers(lect. vyb)**

**num0 = list('111101101101111')**

**num1 = list('001001001001001')**

**num2 = list('111001111100111')**

**num3 = list('111001111001111')**

**num4 = list('101101111001001')**

**num5 = list('111100111001111')**

**num6 = list('111100111101111')**

**num7 = list('111001001001001')**

**num8 = list('111101111101111')**

**num9 = list('111101111001111')**

**nums = [num0, num1, num2, num3, num4, num5, num6, num7, num8, num9]**

**#Five(test)**

**num51 = ('111100111000111')**

**num52 = ('111100010001111')**

**num53 = ('111100011001111')**

**num54 = ('110100111001111')**

**num55 = ('110100111001011')**

**num56 = ('111100101001111')**

**#weight**

**weight = []**

**for i in range(15):**

**weight.append(0)**

**#weght = [0 for i in range(15)] <- то же самое**

**#порог ф-и активации**

**bias = 7**

**#function ob die numer funf ist oder**

**def proceed(number):**

**#взвешенная сумма**

**net = 0**

**for i in range(15):**

**net += weight[i]\*int(number[i])**

**if (net >= bias):**

**return True**

**else:**

**return False**

**#return net>= bias <- nj ;t cfvjt**

**#уменьшение значений весов, если сеть ошиблась и выдала единицу**

**def decrease(number):**

**for i in range(15):**

**if(int(number[i])==1):**

**weight[i] -= 1**

**#увеличение значений весов, если сеть ошиблась и выдала ноль**

**def increase(number):**

**for i in range(15):**

**if(int(number[i])==1):**

**weight[i] += 1**

**#Traning**

**for i in range(100000):**

**option = random.randint(0,9)**

**proceed(nums[option])**

**if(proceed != 5):**

**if (proceed(nums[option])):**

**decrease()**

**else:**

**if not proceed(num5):**

**increase(num5)**

**#Outout**

**print(weight)**

**#test**

**print("0 ist funf?", proceed(num0))**

**print("1 ist funf?", proceed(num1))**

**print("2 ist funf?", proceed(num2))**

**print("3 ist funf?", proceed(num3))**

**print("4 ist funf?", proceed(num4))**

**print("5 ist funf?", proceed(num5))**

**print("6 ist funf?", proceed(num6))**

**print("7 ist funf?", proceed(num7))**

**print("8 ist funf?", proceed(num8))**

**print("9 ist funf?", proceed(num9), "\n")**

**#lernen**

**print("5 - 5?", proceed(num5))**

**print("5 - corrupted 5.1?", proceed(num51))**

**print("5 - corrupted 5.2?", proceed(num52))**

**print("5 - corrupted 5.3?", proceed(num53))**

**print("5 - corrupted 5.4?", proceed(num54))**

**print("5 - corrupted 5.5?", proceed(num55))**

**print("5 - corrupted 5.6?", proceed(num56))**

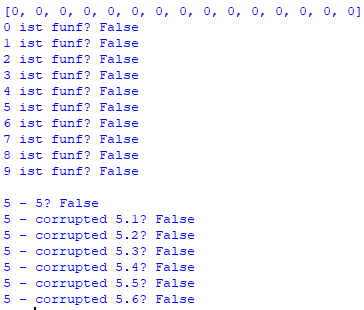
Предоставленные программе учебные данные

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 001001001001001 | | |  | 111001111100111 | | |  | 111001111001111 | | |  | 101101111001001 | | |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 111100111001111 | | |  | 111100111101111 | | |  | 111001001001001 | | |  | 111101111101111 | | |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 111101111001111 | | |  | 111101101101111 | | |  |  |  |  |  |  |  |  |  |

Предоставленные программе тестовые данные

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 111100111000111 | | |  | 111100010001111 | | |  | 111100011001111 | | |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 110100111001111 | | |  | 110100111001011 | | |  | 111100101001111 | | |  |

Результаты выполнения программы:



Результаты работы: мы научились создавать простую перцептронную нейронную сеть и применять для ее обучения правило Хоба