Universidade Federal de Uberlândia
Faculdade de Engenharia Elétrica

Aprendizagem de Máquina

Tarefa 03 - Perceptrons: Reconhecimento de Letras

Docente:

Prof. Keiji Yamanaka

Discente:

Augusto Soares Porto - 12121ECP016

Uberlândia
1 de setembro de 2024

Sumário

Sumário

1 Introdução

Perceptrons são um tipo de rede neural artificial capaz de aprender a classificar corretamente as letras a partir de um conjunto de dados de treinamento, analisando o impacto de diferentes parâmetros, como o peso das conexões e o valor de bias, na eficácia do reconhecimento.

Este relatório foca no estudo e implementação de um algoritmo de perceptron para o reconhecimento de letras do alfabeto, o qual pode ser treinado para diferenciar as 26 letras do alfabeto, cada uma representada por uma matriz de pixels 5x5.

2 Código realizado

2.1 reconhecimento_de_letras_por_perceptron.py

```
from dicionario_5x5 import letras


def main():
    # Treinar o perceptron
    taxa_aprendizado = 0.05 #    adotado
    w, b = algoritmo_perceptron(letras, taxa_aprendizado)

    # Testar com uma entrada nova
    x = [-1, -1, 1, -1, -1,
             -1, 1, -1, -1, -1,
             1, -1, -1, -1, 1,
             1, -1, -1, 1, 1,
             1, -1, -1, -1, 1]

    y = teste_perceptron(x, w, b)

    # Exibir o resultado do perceptron
    estimativa = None

    for i, letra in enumerate(letras.keys()):
        print(f"Saída para a letra {letra}: y = {y[i]}")
        if y[i] == 1:
            estimativa = letras[letra]
```

```
25
26      print("Entrada:")
27      print_letra(x)
28
29      if estimativa is None:
30          print("Não foi possível identificar a letra")
31      else:
32          print("Saída estimada:")
33          print_letra(estimativa)
34
35
36  # Função para imprimir a letra
37  def print_letra(letra):
38      for i in range(5):
39          for j in range(5):
40              print('#' if letra[i*5 + j] == 1 else ' ', end='')
41          print()
42
43
44  # Função de ativação
45  def ativacao(yliq):
46      if yliq >= 0:  # adotei    = 0
47          return 1
48      else:
49          return -1
50
51
52  # Função para treinar o perceptron
53  def algoritmo_perceptron(letras, taxa_aprendizado):
54      num_letras = len(letras)
55      w = [[0] * 25 for _ in range(num_letras)]
56      b = [0] * num_letras
57      alfa = taxa_aprendizado
58
59      while True:
60          pesos_mudaram = False
61
62          for i, (letra, valores) in enumerate(letras.items()):
63              x = valores
64
65              t = []
```

```
66              for j in range(num_letras):
67                  if j == i:
68                      t.append(1)
69                  else:
70                      t.append(-1)
71
72              for j in range(num_letras):
73                  yliq = sum(w[j][k] * x[k] for k in range(25)) + b[j
                        ]
74                  y = ativacao(yliq)
75
76                  if y != t[j]:
77                      pesos_mudaram = True
78                      for k in range(25):
79                          w[j][k] += alfa * x[k] * t[j]
80                      b[j] += alfa * t[j]
81
82          if not pesos_mudaram:
83              break
84
85      return w, b
86
87
88  # Função para testar o perceptron
89  def teste_perceptron(x, w, b):
90      y = []
91      for i in range(len(w)):
92          yliq = sum(w[i][k] * x[k] for k in range(25)) + b[i]
93          y.append(ativacao(yliq))
94      return y
95
96
97  if __name__ == '__main__':
98      main()
```

Listing 1 – Exemplo de código Python

2.2 dicionario$_5x5.py$

Aqui está um exemplo de código Python:

```
# Definição das letras como listas de 25 entradas (5x5)
letras = {
      'A': [-1, -1, 1, -1, -1,
            -1, 1, -1, 1, -1,
            1, -1, -1, -1, 1,
            1, 1, 1, 1, 1,
            1, -1, -1, -1, 1],

      'B': [1, 1, 1, 1, -1,
            1, -1, -1, -1, 1,
            1, 1, 1, 1, -1,
            1, -1, -1, -1, 1,
            1, 1, 1, 1, -1],

      'C': [-1, 1, 1, 1, 1,
            1, -1, -1, -1, -1,
            1, -1, -1, -1, -1,
            1, -1, -1, -1, -1,
            -1, 1, 1, 1, 1],

      'D': [1, 1, 1, 1, -1,
            1, -1, -1, -1, 1,
            1, -1, -1, -1, 1,
            1, -1, -1, -1, 1,
            1, 1, 1, 1, -1],

      'E': [1, 1, 1, 1, 1,
            1, -1, -1, -1, -1,
            1, 1, 1, 1, -1,
            1, -1, -1, -1, -1,
            1, 1, 1, 1, 1],

      'F': [1, 1, 1, 1, 1,
            1, -1, -1, -1, -1,
            1, 1, 1, 1, -1,
            1, -1, -1, -1, -1,
            1, -1, -1, -1, -1],

      'G': [-1, 1, 1, 1, 1,
```

```
            1, 1, -1, -1, -1,
            1, -1, -1, 1, 1,
            1, 1, -1, -1, 1,
            -1, 1, 1, 1, -1],

    'H': [1, -1, -1, -1, 1,
            1, -1, -1, -1, 1,
            1, 1, 1, 1, 1,
            1, -1, -1, -1, 1,
            1, -1, -1, -1, 1],

    'I': [-1, -1, 1, -1, -1,
            -1, -1, 1, -1, -1,
            -1, -1, 1, -1, -1,
            -1, -1, 1, -1, -1,
            -1, -1, 1, -1, -1],

    'J': [-1, -1, -1, 1, -1,
            -1, -1, -1, 1, -1,
            -1, -1, -1, 1, -1,
            -1, 1, -1, 1, -1,
            -1, -1, 1, -1, -1],

    'K': [1, -1, -1, -1, 1,
            1, -1, -1, 1, -1,
            1, 1, 1, -1, -1,
            1, -1, -1, 1, -1,
            1, -1, -1, -1, 1],

    'L': [1, -1, -1, -1, -1,
            1, -1, -1, -1, -1,
            1, -1, -1, -1, -1,
            1, -1, -1, -1, -1,
            1, 1, 1, 1, 1],

    'M': [1, -1, -1, -1, 1,
            1, 1, -1, 1, 1,
            1, -1, 1, -1, 1,
            1, -1, -1, -1, 1,
            1, -1, -1, -1, 1],

```

```
        'N': [1, -1, -1, -1, 1,
              1, 1, -1, -1, 1,
              1, -1, 1, -1, 1,
              1, -1, -1, 1, 1,
              1, -1, -1, -1, 1],

        'O': [-1, 1, 1, 1, -1,
              1, -1, -1, -1, 1,
              1, -1, -1, -1, 1,
              1, -1, -1, -1, 1,
              -1, 1, 1, 1, -1],

        'P': [1, 1, 1, 1, -1,
              1, -1, -1, 1, 1,
              1, 1, 1, 1, -1,
              1, -1, -1, -1, -1,
              1, -1, -1, -1, -1],

        'Q': [-1, 1, 1, 1, -1,
              1, -1, -1, -1, 1,
              1, -1, 1, -1, 1,
              1, -1, 1, -1, 1,
              -1, 1, 1, 1, -1],

        'R': [1, 1, 1, 1, -1,
              1, -1, -1, -1, 1,
              1, 1, 1, 1, -1,
              1, -1, -1, 1, -1,
              1, -1, -1, -1, 1],

        'S': [-1, 1, 1, 1, 1,
              1, -1, -1, -1, -1,
              -1, 1, 1, 1, -1,
              -1, -1, -1, -1, 1,
              1, 1, 1, 1, -1],

        'T': [1, 1, 1, 1, 1,
              -1, -1, 1, -1, -1,
              -1, -1, 1, -1, -1,
              -1, -1, 1, -1, -1,
              -1, -1, 1, -1, -1],
```

```
122
123         'U': [1, -1, -1, -1, 1,
124               1, -1, -1, -1, 1,
125               1, -1, -1, -1, 1,
126               1, -1, -1, -1, 1,
127               -1, 1, 1, 1, -1],
128
129         'V': [1, -1, -1, -1, 1,
130               1, -1, -1, -1, 1,
131               -1, 1, -1, 1, -1,
132               -1, 1, -1, 1, -1,
133               -1, -1, 1, -1, -1],
134
135         'X': [1, -1, -1, -1, 1,
136               -1, 1, -1, 1, -1,
137               -1, -1, 1, -1, -1,
138               -1, 1, -1, 1, -1,
139               1, -1, -1, -1, 1],
140
141         'Y': [1, -1, -1, -1, 1,
142               -1, 1, -1, 1, -1,
143               -1, -1, 1, -1, -1,
144               -1, -1, 1, -1, -1,
145               -1, -1, 1, -1, -1],
146
147         'Z': [1, 1, 1, 1, 1,
148               -1, -1, -1, 1, -1,
149               -1, -1, 1, -1, -1,
150               -1, 1, -1, -1, -1,
151               1, 1, 1, 1, 1]
152 }
```

Listing 2 – Exemplo de código Python

3 Saída do código realizado

Vale ressaltar que a saída não corresponde a matriz com os 26 neurônios, mas sim a saída ao neurônio correspondente a letra determinada.

3.1 Teste 1

```
x = [-1, -1, 1, -1, -1,
     -1, 1, -1, 1, -1,
     1, -1, -1, -1, 1,
     1, 1, 1, 1, 1,
     1, -1, -1, -1, 1]
```

```
Saída para a letra A: y = 1
Saída para a letra B: y = -1
Saída para a letra C: y = -1
Saída para a letra D: y = -1
Saída para a letra E: y = -1
Saída para a letra F: y = -1
Saída para a letra G: y = -1
Saída para a letra H: y = -1
Saída para a letra I: y = -1
Saída para a letra J: y = -1
Saída para a letra K: y = -1
Saída para a letra L: y = -1
Saída para a letra M: y = -1
Saída para a letra N: y = -1
Saída para a letra O: y = -1
Saída para a letra P: y = -1
Saída para a letra Q: y = -1
Saída para a letra R: y = -1
Saída para a letra S: y = -1
Saída para a letra T: y = -1
Saída para a letra U: y = -1
Saída para a letra V: y = -1
Saída para a letra X: y = -1
Saída para a letra Y: y = -1
Saída para a letra Z: y = -1
Entrada:
  #
 # #
#   #
#####
#   #
Saída estimada:
  #
 # #
#   #
```

```
    #####
    #    #
```

3.2 Teste 2

```
    x = [-1, -1, 1, -1, -1,
         -1, 1, -1, -1, -1,
         1, -1, -1, -1, 1,
         1, -1, -1, 1, 1,
         1, -1, -1, -1, 1]
```

```
    Saída para a letra A: y = 1
    Saída para a letra B: y = -1
    Saída para a letra C: y = -1
    Saída para a letra D: y = -1
    Saída para a letra E: y = -1
    Saída para a letra F: y = -1
    Saída para a letra G: y = -1
    Saída para a letra H: y = -1
    Saída para a letra I: y = -1
    Saída para a letra J: y = -1
    Saída para a letra K: y = -1
    Saída para a letra L: y = -1
    Saída para a letra M: y = -1
    Saída para a letra N: y = -1
    Saída para a letra O: y = -1
    Saída para a letra P: y = -1
    Saída para a letra Q: y = -1
    Saída para a letra R: y = -1
    Saída para a letra S: y = -1
    Saída para a letra T: y = -1
    Saída para a letra U: y = -1
    Saída para a letra V: y = -1
    Saída para a letra X: y = -1
    Saída para a letra Y: y = -1
    Saída para a letra Z: y = -1
    Entrada:
      #
     #
    #    #
    #   ##
```

```
31      #   #
32      Saída estimada:
33        #
34       # #
35      #   #
36      #####
37      #   #
```

3.3 Teste 2

```
1       x = [1, 1, 1, 1, 1,
2            1, -1, -1, -1, 1,
3            1, 1, 1, -1, -1,
4            -1, -1, -1, 1, -1,
5            1, -1, -1, -1, 1]
```

```
1       Saída para a letra A: y = -1
2       Saída para a letra B: y = -1
3       Saída para a letra C: y = -1
4       Saída para a letra D: y = -1
5       Saída para a letra E: y = -1
6       Saída para a letra F: y = -1
7       Saída para a letra G: y = -1
8       Saída para a letra H: y = -1
9       Saída para a letra I: y = -1
10      Saída para a letra J: y = -1
11      Saída para a letra K: y = -1
12      Saída para a letra L: y = -1
13      Saída para a letra M: y = -1
14      Saída para a letra N: y = -1
15      Saída para a letra O: y = -1
16      Saída para a letra P: y = -1
17      Saída para a letra Q: y = -1
18      Saída para a letra R: y = 1
19      Saída para a letra S: y = -1
20      Saída para a letra T: y = -1
21      Saída para a letra U: y = -1
22      Saída para a letra V: y = -1
23      Saída para a letra X: y = -1
24      Saída para a letra Y: y = -1
25      Saída para a letra Z: y = -1
```

```
Entrada:
#####
#   #
###
   #
#   #
Saída estimada:
####
#   #
####
#  #
#   #
```

4 Referências

- Link do diretório com código feito: https://github.com/AugustoSoaresPorto/amaq-ufu
- Link do material de apoio: aqui