

Julio Cesar Nunes

Possui graduação em Ciências Biológicas, Mestrado em Oncologia, com experiência em Bioinformática aplicada à investigação em Câncer. Título da Tese: Análise correlacional entre a expressão dos fatores de splicing e a ocorrência de splicing alternativo em tecidos humanos e de camundongos.

Doutorado em andamento em Bioinformática - Universidade de São Paulo, USP. Orientador: Paulo Sérgio Lopes de Oliveira, HCFMUSP - INCOR - SP.

[Currículo Lattes](#)

Lista de exercícios resolvidos:

- [Exercício 1](#)
- [Exercício 2](#)
- [Exercício 3](#)
- [Exercício 4](#)
- [Exercício 5](#)
- [Exercício 6](#)
- [Exercício 7](#)
- [Exercício 8](#)
- [Exercício 9](#)

Proposta de Trabalho Final

Principal

Redes neurais artificiais

As redes neurais artificiais são aplicadas principalmente nos problemas que envolvem o reconhecimento e a classificação de padrões. A proposta deste trabalho possui como objetivo principal fazer a implementação de uma função com propriedade de reconhecimento de padrão. A função efetuará o reconhecimento através dos seguintes métodos:

- A partir de uma matriz de entrada, a rede é treinada pela função de aprendizagem adotada.
- O modelo implementado possui sua regra de treinamento, onde os pesos de suas conexões são ajustados de acordo com os padrões apresentados.
- O aprendizado ocorre através de exemplos de entrada.

- No momento de testes as matrizes de entradas serão alteradas pelo usuário, no intuito de se verificar seu reconhecimento pela função da rede neural.
- A matriz de entrada, alterada em fase de teste, deverá ser reconhecida pela função.
- A saída será a indicação da matriz original correspondente à modificada.

Plano B

Uma árvore com raiz, na qual cada nó tem dois descendentes, é chamada de árvore binária. O plano B apresenta a idéia de gerar uma função que desenhe árvores binárias. Um conjunto de dados de entrada, produz um arquivo de saída estruturado em uma árvore binária, em formato PostScript, que pode ser impresso na maioria das impressoras e visualizado na maioria dos terminais.

Comentário

O plano A parece um desafio enorme, preocupa-me a viabilidade de executá-lo em tão pouco tempo. Imagino que tenha o ferramental necessário para implementá-la, mas mesmo assim é um bruta desafio. Pondere sobre o tempo disponível e caso ache que não dá conta, opte pelo plano B. Eu realmente fiquei muito curiosos para ver a função A, apesar, ou devido a minha ignorância sobre o assunto ...

— [Alexandre Adalardo de Oliveira](#) 2010/03/31 16:16

Página de Ajuda

neural.r

package:unknown

R Documentation

Aplicação de rede neural artificial no reconhecimento de 6 caracteres alfabéticos dispostos em matrizes binárias.

Description:

A função com propriedades de rede neural artificial efetua reconhecimento padrão de uma letra em formato matricial binário, através da entrada de 6 vetores binários dispostos de modo a representarem as respectivas letras de interesse.

O modelo implementado possui sua regra de treinamento, onde os pesos de suas conexões são ajustados de acordo com os padrões apresentados.

Usage:

```
neural(input, vetor1=letra.a, vetor2=letra.b, vetor3=letra.c,  
vetor4=letra.d, vetor5=letra.e, vetor6=letra.k, alvo1=target.a,  
alvo2=target.b, alvo3=target.c, alvo4=target.d, alvo5=target.e,  
alvo6=target.k, rows_pat=9, cols_pat=7, rows_mini=5, cols_mini=3)
```

Arguments:

input: Vetor numérico. Valores advindos da matriz binária investigada.

vetor1: Vetor numérico. Valores advindos da primeira matriz binária padrão.

vetor2: Vetor numérico. Valores advindos da segunda matriz binária padrão.

vetor3: Vetor numérico. Valores advindos da terceira matriz binária padrão.

vetor4: Vetor numérico. Valores advindos da quarta matriz binária padrão.

vetor5: Vetor numérico. Valores advindos da quinta matriz binária padrão.

vetor6: Vetor numérico. Valores advindos da sexta matriz binária padrão.

alvo1: Vetor numérico. Valores advindos da primeira matriz binária de treinamento.

alvo2: Vetor numérico. Valores advindos da segunda matriz binária de treinamento.

alvo3: Vetor numérico. Valores advindos da terceira matriz binária de treinamento.

alvo4: Vetor numérico. Valores advindos da quarta matriz binária de treinamento.

alvo5: Vetor numérico. Valores advindos da quinta matriz binária de treinamento.

alvo6: Vetor numérico. Valores advindos da sexta matriz binária de treinamento.

rows_pat: Numérico. Número desejado de linhas nas matrizes padrão e investigada.

cols_pat: Numérico. Número desejado de colunas nas matrizes padrão e investigada.

`rows_mini`: Numérico. Número desejado de linhas nas matrizes de treinamento.

`cols_mini`: Numérico. Número desejado de colunas nas matrizes de treinamento.

Details:

O aprendizado da rede neural artificial ocorre através de exemplos de entrada.

No momento de testes os valores binários do vetor de entrada, a ser investigado, pode ser alterado pelo usuário, no intuito de se verificar seu reconhecimento pela função.

A saída será a indicação da matriz original correspondente à modificada, mas com dimensão de tamanho de matriz inferior.

A traço das letras dispostas em matrizes é seguido pelo valor numérico 1.

A função possui a propriedade funcional de reconhecer outros padrões matriciais de interesse.

Warning:

As matrizes padrão e investigada devem possuir mesma dimensão de tamanho, bem como as matrizes de treinamento de dimensão inferior.

Author(s):

Julio Cesar Nunes

nunesjulioc@usp.br

References:

Fausett, V.L. (1994) Fundamentals of Neural Networks: Architectures, Algorithms And Applications, Prentice Hall.

Examples:

```
## Buscando o reconhecimento da letra "A" dentre outras 5 possíveis:
## Letra "A" (Vetor numérico. Valores advindos da matriz binária a ser
investigada).
## A saída deve apresentar uma matriz binária de menor dimensão mas
representativa a letra "A"

input =
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,-
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,-1,-1,1,-
1,-1,1,-1,-1,-1,1,-1)

## Buscando o reconhecimento da letra "K" dentre outras 5 possíveis:
## Letra "K" (Vetor numérico. Valores advindos da matriz binária a ser
investigada).
## A saída deve apresentar uma matriz binária de menor dimensão mas
representativa a letra "K"

input =
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,-
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,-1,-
1,1,-1,-1,-1,-1,1,-1)

## Exemplos de 6 vetores numéricos. Valores advindos das 6 matrizes binárias
padrão.
## Letra A:
letra.a =
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,-
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,-1,-1,-1,1,-
1,-1,1,-1,-1,-1,1,-1)
## Letra B:
letra.b =
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,-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,-1,-1,1,1,1,
1,1,1,-1)
## Letra C:
letra.c =
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,-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,
,-1,-1,-1,1,1,1,-1,-1)
## Letra D:
letra.d =
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,-1,1,
-1,1,-1,-1,-1,-1,1,-1,1,-1,-1,-1,-1,1,-1,1,-1,-1,-1,1,-1,-1,-1,1,-1,
-1,1,1,1,1,-1,-1)
## Letra E:
letra.e =
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,-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,-1,-1,
1,1,1,1,1,1,1)
## Letra K:
letra.k =
```

$$c(1, -1, -1, -1, -1, 1, -1, 1, -1, -1, -1, 1, -1, -1, 1, -1, -1, -1, 1, -1, 1, -1, -1, -1, -1, 1, 1, -1, -1, -1, -1, -1, 1, -1, 1, -1, -1, -1, -1, 1, -1, -1, -1, 1, -1, -1, 1, -1, -1, -1, -1, 1, -1, -1, -1, -1, 1, -1)$$

```
## 6 vetores numéricos. Valores advindos das matrizes binárias de
treinamento
## Matrizes de treinamento e saídas esperadas das respectivas letras:
## Letra A:
target.a = c(-1,1,-1,1,-1,1,1,1,1,-1,1,1,-1,1)
## Letra B:
target.b = c(1,1,1,1,-1,1,1,1,-1,1,-1,1,1,1)
## Letra C:
target.c = c(1,1,1,1,-1,1,1,-1,-1,1,-1,1,1,1)
## Letra D:
target.d = c(1,1,-1,1,-1,1,1,-1,1,1,-1,1,1,1)
## Letra E:
target.e = c(1,1,1,1,-1,-1,1,1,-1,1,-1,-1,1,1)
## Letra K:
target.k = c(1,-1,1,1,1,-1,1,-1,-1,1,1,-1,1,-1)
```

```
neural(input, vetor1=letra.a, vetor2=letra.b, vetor3=letra.c,
vetor4=letra.d, vetor5=letra.e, vetor6=letra.k, alvo1=target.a,
alvo2=target.b, alvo3=target.c, alvo4=target.d, alvo5=target.e,
alvo6=target.k, rows pat=9, cols pat=7, rows mini=5, cols mini=3)
```

Código da Função

```
neural <- function(input, vetor1=letra.a, vetor2=letra.b, vetor3=letra.c,
vetor4=letra.d, vetor5=letra.e, vetor6=letra.k, alvo1=target.a,
alvo2=target.b, alvo3=target.c, alvo4=target.d, alvo5=target.e,
alvo6=target.k, rows_pat=9, cols_pat=7, rows_mini=5, cols_mini=3)
{
  in_mat = matrix(input, nrow=rows_pat, ncol=cols_pat, byrow=T)
  in_trans = t(in_mat)
  in_col = matrix(in_trans, ,1)
  in_ftrow = t(in_col)
  mat1 = matrix(vetor1, nrow=rows_pat, ncol=cols_pat, byrow=T)
  mat1_trans = t(mat1)
  mat1_col = matrix(mat1_trans, ,1)
  mat1_ftrow = t(mat1_col)
  n.pat1 = dim(mat1_ftrow)
  minil = matrix(alvo1, nrow=rows_mini, ncol=cols_mini, byrow=T)
  minil_col = matrix(minil, ,1)
  minil_ftrow = t(minil_col)
  n.minil = dim(minil_ftrow)
```

```
w1 = matrix(0,n.pat1[2],n.mini1[2])
for(j in 1:n.mini1[2]) {
  for(i in 1:n.pat1[2])
  {
    w1[i,j]=w1[i,j]+mat1_ftrow[1, i]*mini1_ftrow[1, j];
  }
}
mat2 = matrix(vetor2, nrow=rows_pat, ncol=cols_pat, byrow=T)
mat2_trans = t(mat2)
mat2_col = matrix(mat2_trans, ,1)
mat2_ftrow = t(mat2_col)
n.pat2 = dim(mat2_ftrow)
mini2 = matrix(alvo2, nrow=rows_mini, ncol=cols_mini, byrow=T)
mini2_col = matrix(mini2, ,1)
mini2_ftrow = t(mini2_col)
n.mini2 = dim(mini2_ftrow)
w2 = matrix(0,n.pat2[2],n.mini2[2])
for(j in 1:n.mini2[2]) {
  for(i in 1:n.pat2[2])
  {
    w2[i,j]=w2[i,j]+mat2_ftrow[1, i]*mini2_ftrow[1, j];
  }
}
mat3 = matrix(vetor3, nrow=rows_pat, ncol=cols_pat, byrow=T)
mat3_trans = t(mat3)
mat3_col = matrix(mat3_trans, ,1)
mat3_ftrow = t(mat3_col)
n.pat3 = dim(mat3_ftrow)
mini3 = matrix(alvo3, nrow=rows_mini, ncol=cols_mini, byrow=T)
mini3_col = matrix(mini3, ,1)
mini3_ftrow = t(mini3_col)
n.mini3 = dim(mini3_ftrow)
w3 = matrix(0,n.pat3[2],n.mini3[2])
for(j in 1:n.mini3[2]) {
  for(i in 1:n.pat3[2])
  {
    w3[i,j]=w3[i,j]+mat3_ftrow[1, i]*mini3_ftrow[1, j];
  }
}
mat4 = matrix(vetor4, nrow=rows_pat, ncol=cols_pat, byrow=T)
mat4_trans = t(mat4)
mat4_col = matrix(mat4_trans, ,1)
mat4_ftrow = t(mat4_col)
n.pat4 = dim(mat4_ftrow)
mini4 = matrix(alvo4, nrow=rows_mini, ncol=cols_mini, byrow=T)
mini4_col = matrix(mini4, ,1)
mini4_ftrow = t(mini4_col)
n.mini4 = dim(mini4_ftrow)
w4 = matrix(0,n.pat4[2],n.mini4[2])
for(j in 1:n.mini4[2]) {
  for(i in 1:n.pat4[2])
```

```
{
    w4[i,j]=w4[i,j]+mat4_ftrow[1, i]*mini4_ftrow[1, j];
}
}
mat5 = matrix(vetor5, nrow=rows_pat, ncol=cols_pat, byrow=T)
mat5_trans = t(mat5)
mat5_col = matrix(mat5_trans, ,1)
mat5_ftrow = t(mat5_col)
n.pat5 = dim(mat5_ftrow)
mini5 = matrix(alvo5, nrow=rows_mini, ncol=cols_mini, byrow=T)
mini5_col = matrix(mini5, ,1)
mini5_ftrow = t(mini5_col)
n.mini5 = dim(mini5_ftrow)
w5 = matrix(0,n.pat5[2],n.mini5[2])
for(j in 1:n.mini5[2]) {
    for(i in 1:n.pat5[2])
    {
        w5[i,j]=w5[i,j]+mat5_ftrow[1, i]*mini5_ftrow[1, j];
    }
}
mat6 = matrix(vetor6, nrow=rows_pat, ncol=cols_pat, byrow=T)
mat6_trans = t(mat6)
mat6_col = matrix(mat6_trans, ,1)
mat6_ftrow = t(mat6_col)
n.pat6 = dim(mat6_ftrow)
mini6 = matrix(alvo6, nrow=rows_mini, ncol=cols_mini, byrow=T)
mini6_col = matrix(mini6, ,1)
mini6_ftrow = t(mini6_col)
n.mini6 = dim(mini6_ftrow)
w6 = matrix(0,n.pat6[2],n.mini6[2])
for(j in 1:n.mini6[2]) {
    for(i in 1:n.pat6[2])
    {
        w6[i,j]=w6[i,j]+mat6_ftrow[1, i]*mini6_ftrow[1, j];
    }
}
w = w1+w2+w3+w4+w5+w6
y_in = matrix(0, ,n.mini1[2])
y_out = matrix(0, ,n.mini1[2])
for(j in 1:n.mini1[2]) {
    for(i in 1:n.pat1[2]) {
        y_in[j]=y_in[j]+in_ftrow [1, i]*w[i,j]
        if (y_in[j]>0) {
            y_out[j] = 1
        } else {
            y_out[j] = -1
        }
    }
}
y = y_out
```



```
matriz = matrix(y,5,3)
return(matriz)
}
```

Arquivo da Função

[neural.r](#)

From:

<http://ecor.ib.usp.br/> - **ecoR**

Permanent link:

http://ecor.ib.usp.br/doku.php?id=05_curso_antigo:r2010:alunos:trabalho_final:juliocnunes:start 

Last update: **2020/08/12 06:04**