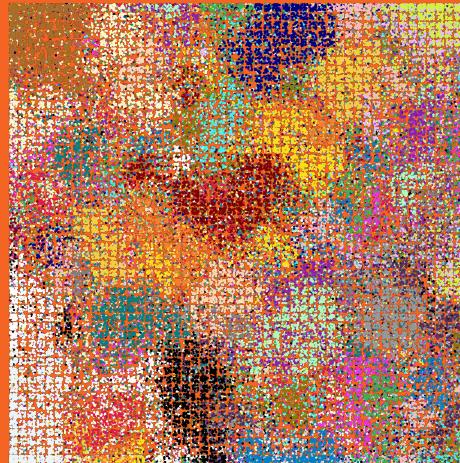


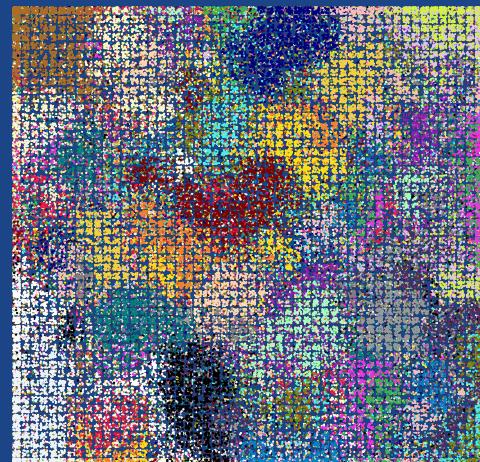
Self Organizing Maps



Oficina: Neuremberg de Matos, Ricardo Gontijo, Stefano Dantas

Plano de apresentação

- Introdução
 - História
 - Breve explicação
 - Aplicação
- Explicação matemática
- Aplicação Python



Introdução

● História

Self-Organizing Maps foi uma contribuição do acadêmico finlandês Dr. Teuvo Kohonen.

- 1981 IBM Personal Computer
- 1982 Self-Organizing Maps
- 1984 Apple Macintosh
- 1985 Microsoft Windows

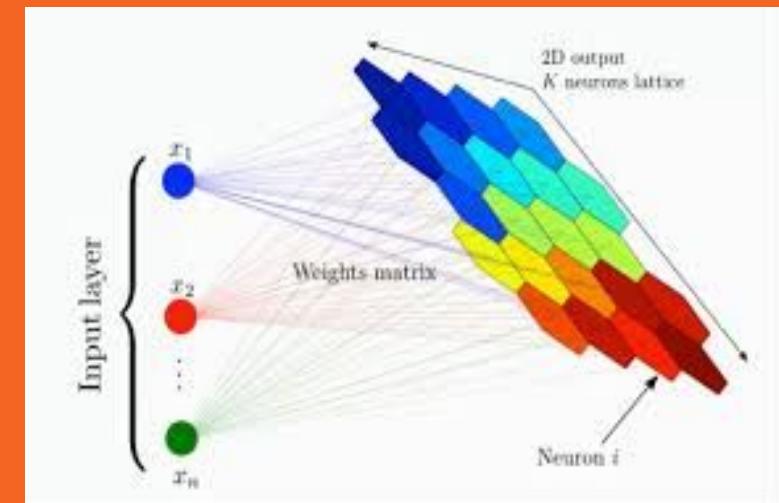


Introdução

- Breve explicação

Self-Organizing Maps:

- Modelo de rede neural não supervisionado.
- São usados para criar uma representação ordenada de dados multidimensionais que simplifica a complexidade e revela relações significativas. Destinado a aplicações em que a manutenção de uma topologia entre os espaços de entrada e saída é importante.

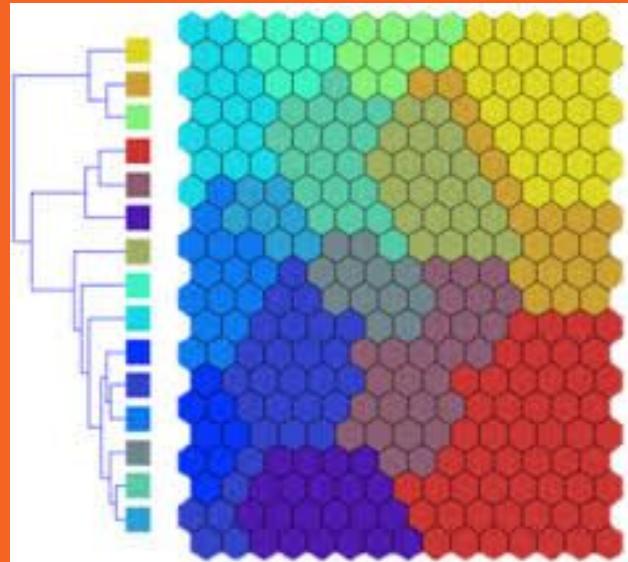


Introdução

- Breve explicação

Self-Organizing Maps:

- A característica do algoritmo é que os vetores de entrada que são próximos - semelhantes - no espaço dimensional alto também são mapeados para nós próximos no espaço 2D. É um método para redução de dimensionalidade.
- Por ser um modelo não supervisionado, os nós são auto-organizados.
- São chamados de “feature maps”, pelo fato de estarem retreinando as características dos dados de entrada e simplesmente agrupando-os de acordo com a semelhança.



Introdução

● Aplicação

Possíveis áreas encontradas para aplicação de SOM foram:

→ Meteorologia

- ◆ Evaporação, precipitação (chuva e neve) e classificação de nuvem com base em observações in situ, modelo de saída e imagens de satélite. Muitas dessas aplicações também são encontradas no campo da hidrologia.
- ◆ Usado para resumir e descrever o padrões de circulação atmosférica indicados pela pressão ao nível do mar e altura geopotencial em diferentes níveis, e relacionar os padrões de circulação característicos com outros.
- ◆ Examinar os padrões de covariabilidade entre diversas variáveis meteorológicas, tais como: temperatura do ar, umidade, and dados de vento.

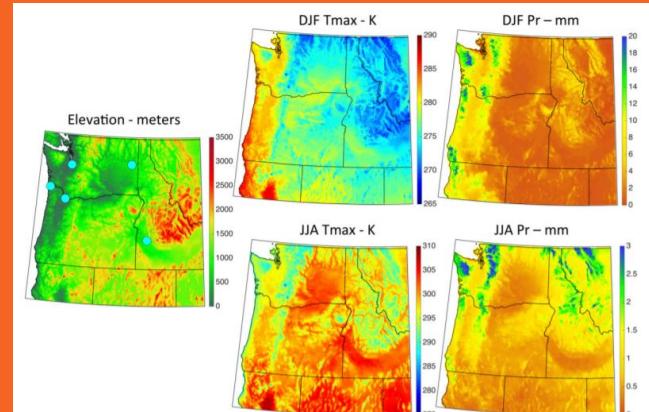


FIG. 1. (left) Elevation and (middle) mean daily maximum temperature and (right) mean daily precipitation for (top) DJF and (bottom) JJA for the NWUS region. Note the changes in color scale between DJF and JJA. (Blue dots on the elevation map are locations of local-scale examples in Figs. 10 and 15.)

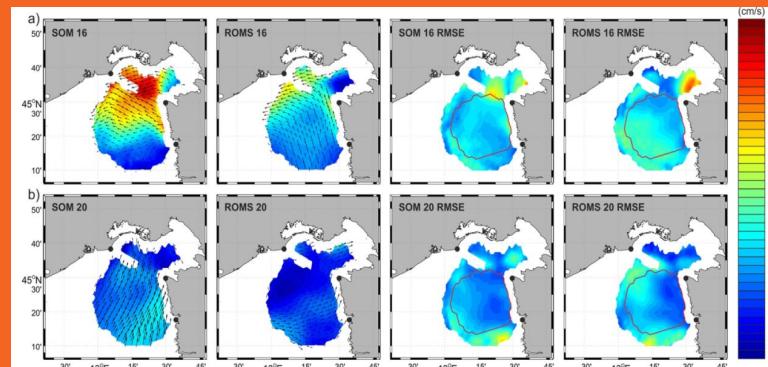
Introdução

● Aplicação

Possíveis áreas encontradas para aplicação de SOM foram:

→ Oceanografia

- ◆ Dados de satélite de temperatura da superfície do mar, de altura da superfície do mar, imagens de satélite de cor do oceano e clorofila.
- ◆ Além disso, aplicações de SOM foram encontradas em muitos dados oceanográficos, como estresse do vento, formato do fundo do mar, tsunami e salinidade.



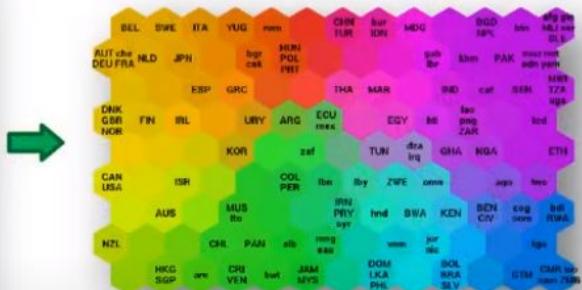
Introdução

● Aplicação

Possíveis áreas encontradas para aplicação de SOM foram:

→ Visualização diversas

A	B	C	D	E	
1	Country	Country C	Health	Education	Inflation
2	Aruba	ABW	9.418971	5.92467022	-2.13637
3	Afghanistan	AFG	4.371774		-8.28308
4	Angola	AGO	5.791339		13.773145
5	Albania	ALB	6.75969		2.280502
6	Andorra	AND	4.57058	3.1638701	
7	Arab World	ARB	4.049924		3.524814
8	United Arab Emirates	ARE	7.634758		
9	Argentina	ARG	4.545323	4.88997984	6.282774
10	Armenia	ARM		3.84079003	3.406767
11	American Samoa	ASM	4.862062		
12	Antigua and Barbuda	ATG	9.046056	2.55447006	-0.55016
13	Australia	AUS	11.19444	5.09262991	1.820112
14	Austria	AUT	5.85024	5.7674098	0.506313
15	Azerbaijan	AZE	6.964187	3.22430992	1.401056
16	Burundi	BDI	10.39434	6.3197999	10.98147
17	Belgium	BEL	4.46431	6.41535997	-0.05315
18	Benin	BEN	7.405431	4.22204018	2.15683



Intuição Matemática

- Dado um *input* multidimensional, produzir uma representação desses dados em um espaço bidimensional (o "mapa")
- Os dados não têm classes pré definidas, queremos encontrar um algoritmo que seja capaz de se "auto-organizar"
- Formalmente, queremos achar uma função f tal que:

$$f : \mathcal{X} \rightarrow \mathcal{Y}, \mathcal{X} \in \mathbb{R}^D, \mathcal{Y} \in \mathbb{R}^2$$

Intuição Matemática

- Uma rede neural é um aproximador de funções extremamente versátil e poderoso.
- Problema: como treinar sem classes?
- Em vez de usar *backpropagation*, a rede é treinada usando aprendizado competitivo

Algoritmo

$$\mathbf{X} = \{X_1, X_2, \dots, X_N\}, \quad \mathbf{W} = \{W_1, W_2, \dots, W_M\}, W_i \in \mathbb{R}^D \forall i$$
$$X_i = \{x_{i,1}, x_{i,2}, \dots, x_{i,D}\}$$

1 - Inicializar neurônios com pesos aleatórios

2- Selecionar uma entrada qualquer X_i

Algoritmo

$$\mathbf{X} = \{X_1, X_2, \dots, X_N\}, \quad \mathbf{W} = \{W_1, W_2, \dots, W_M\}, W_i \in \mathbb{R}^D \forall i$$
$$X_i = \{x_{i,1}, x_{i,2}, \dots, x_{i,D}\}$$

1 - Inicializar neurônios com pesos aleatórios

2- Selecionar uma entrada qualquer X_i

3- Calcular a distância entre X_i e todos os pesos \mathbf{W} , o peso com menor distância será selecionado como *best matching unit* (BMU) $L(X)$

$$L(X) = W_k, k = \operatorname{argmin}_j \|X - W_j\|_2$$

Algoritmo

$$\mathbf{X} = \{X_1, X_2, \dots, X_N\}, \quad \mathbf{W} = \{W_1, W_2, \dots, W_M\}, W_i \in \mathbb{R}^D \forall i$$
$$X_i = \{x_{i,1}, x_{i,2}, \dots, x_{i,D}\}$$

1 - Inicializar neurônios com pesos aleatórios

2- Selecionar uma entrada qualquer X_i

3- Calcular a distância entre X_i e todos os pesos \mathbf{W} , o peso com menor distância será selecionado como *best matching unit* (BMU) $L(X)$

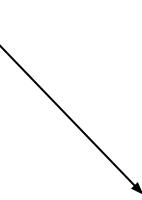
4- Atualizar os pesos usando:

$$W_v(n+1) = W_v(n) + \eta(n)h_{j,L(x)}(n)(X_i - W_v)$$

5- $n+1$, repetir a partir do passo 2 até o número de iterações desejado

Atualização dos pesos

$$\Delta W = \eta(n) h_{j,L(x)}(n) (X_i - W_v)$$



taxa de
aprendizado

$$\eta(n) = \eta_0 \exp\left(\frac{-n}{\tau_\eta}\right)$$

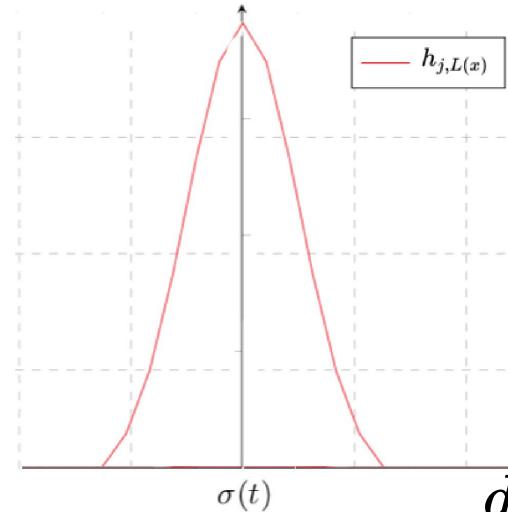
Atualização dos pesos

$$\Delta W = \eta(n) h_{j,L(x)}(n) (X_i - W_v)$$

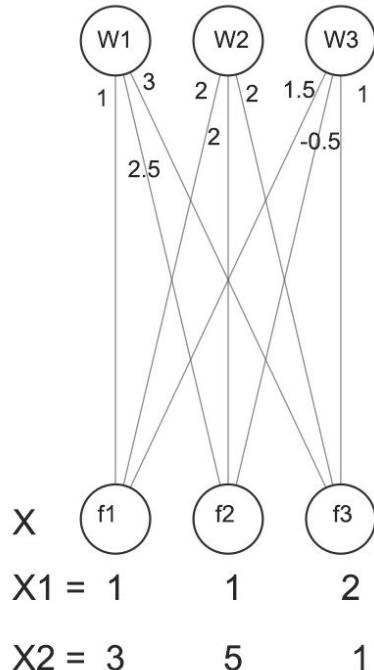
função de
vizinhança

$$h_{j,L(x)}(n) = \exp\left(\frac{-d_{j,L(x)}^2}{2\sigma(n)^2}\right)$$

$$\sigma(n) = \exp\left(\frac{-n}{\tau_0}\right)$$



Exemplo



Selecionamos X_1

$$d_{X_1, W_1} = \sqrt{(1 - 1)^2 + (2.5 - 1)^2 + (3 - 2)^2} = 1.8$$

$$d_{X_1, W_2} = \sqrt{(2 - 1)^2 + (2 - 1)^2 + (2 - 2)^2} = 1.41$$

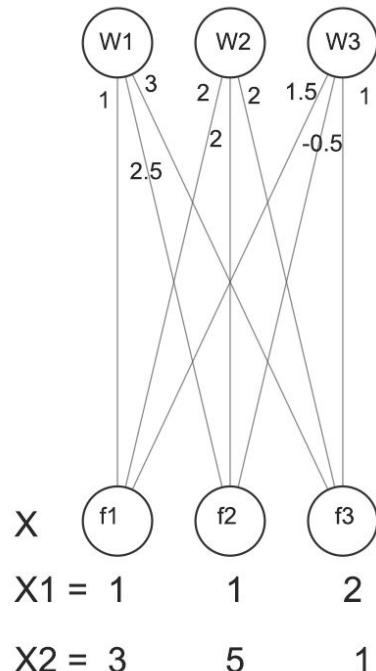
$$d_{X_1, W_3} = \sqrt{(1.5 - 1)^2 + (-0.5 - 1)^2 + (2 - 1)^2} = 1.68$$

W_2 tem a menor distância

$$L(X) = W_2$$

Exemplo

Atualizando os pesos



$$\Delta W_1 = \eta(1)h_{1,2}(1)(X_1 - W_1)$$

$$\Delta W_2 = \eta(1)h_{2,2}(1)(X_1 - W_2)$$

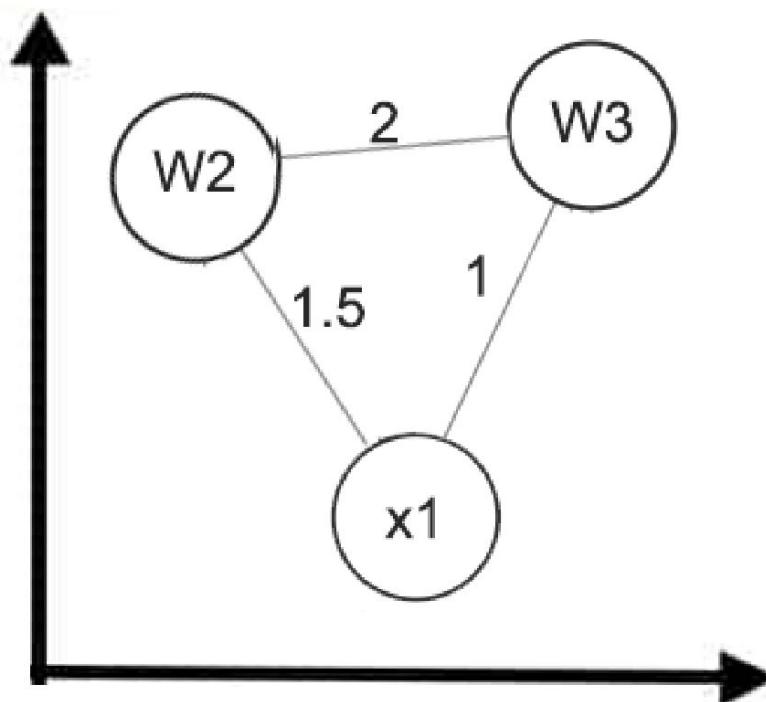
$$\Delta W_3 = \eta(1)h_{3,2}(1)(X_1 - W_3)$$

Representação em 2D

$$d_{X1,W_2} = 1.5$$

$$d_{X1,W_3} = 1$$

$$d_{W2,W_3} = 2$$



Obrigado!

Perguntas?