

1 Neurónové siete

1.1 Kompetičné siete

Máme množinu dvojrozmerných vektorov a chceme ju rozdeliť na skupiny podobných vektorov - nájsť zhluky.

```
>> p=[ 0.1  4  5  0.2  15  3.8  14  ;  
       10  0  1  13   5  0.4  5.9];
```

Vytvoríme kompetičnú sieť s 2 vstupnými a 3 neurónmi v kompetičnej vrstve:

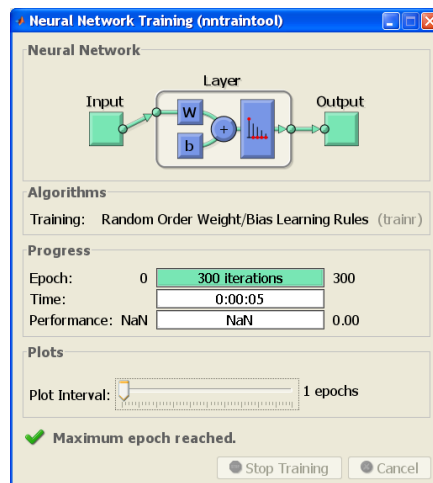
```
>> net=newc([0 15; 0 10],3);
```

Prvý parameter funkcie `newc` udáva rozsahy hodnôt jednotlivých zložiek vstupných vektorov, druhý argument udáva počet neurónov v kompetičnej vrstve. Nastavíme počet iterácií cez vstupné vzory

```
>> net.trainParam.epochs=300;
```

Spustíme tréovanie siete.

```
>> net = train(net,p);
```



Natrénovanú sieť necháme spočítať výstupy pre vzory z `p`:

```
>> a= sim(net,p)
```

a =

```
(3,1)      1  
(2,2)      1  
(2,3)      1  
(3,4)      1  
(1,5)      1  
(2,6)      1  
(1,7)      1
```

Výsledkom je riedka matica – v každom stĺpci je len jedna hodnota 1. Nasledujúca funkcia vráti riadkové indexy prvkov 1 z matice **a**.

```
>> ac=vec2ind(a)
```

```
ac =
```

```
      3      2      2      3      1      2      1
```

To sú vlastne čísla zhlukov, do ktorých bol zaradený príslušný vstupný vektor. Zhluky môžeme zobrazíť farebne rozlíšené. Najprv si uložíme indexy vektorov jednotlivých zhlukov a potom ich zobrazíme rôznymi farbami:

```
>> ind1=find(ac==1)
```

```
ind1 =
```

```
      5      7
```

```
>> ind2=find(ac==2);
```

```
>> ind3=find(ac==3);
```

Nasledujúci príkaz musí byť celý na jedinom riadku!

```
>> plot(p(1,ind1),p(2,ind1),'or',p(1,ind2),p(2,ind2),'og',
        p(1,ind3),p(2,ind3),'ob')
```

1.2 Kohonenove samo-organizujúce sa mapy

Tento model je vlastne rozšírením kompetičnej siete. Neuróny majú navyše definované topológiu. Vytvorenie Kohonenovej mapy:

```
net = newsom(P, [D1,D2,...],TFCN,DFCN,OSTEPS,IN)
```

P matica $R \times Q$ obsahujúca Q vstupných vektorov,

D_i rozmery siete; implicitne = [5 8],

TFCN topológia; implicitne = 'hextop',

DFCN funkcia vzdialenosti, implicitne = 'linkdist',

OSTEPS počet krokov fázy usporiadavania, implicitne = 100,

IN počiatočná veľkosť okolia, implicitne = 1.

Možné topológie sú 'hextop', 'gridtop' a 'randtop'. Funkcia vzdialenosti môže byť 'linkdist', 'dist', 'dist' alebo 'mandist', ktoré zodpovedajú počtu prejdenných hrán, Euklidovskej vzdialenosti, štvorcovému okoliu a Manhattanskej vzdialenosti.

Odkrokuje si demo program Demos → Self-organizing networks → Two-Dimensional Self-organizing Map. Možné topológie si môžete zobrazíť pomocou

```
>> pos=gridtop(2,4)
```

```
pos =
```

0	1	0	1	0	1	0	1
0	0	1	1	2	2	3	3

```
>> plotsom(pos)
```

```
>> pos=hextop(3,4);
```

```
>> plotsom(pos)
```

```
>> pos=randtop(3,4);
```

```
>> plotsom(pos)
```

Učenie v MATLABe má dve fázy. V prvej sa neuróny majú usporiadať podľa topológie a v druhej majú spresniť aproximáciu vstupných vzorov. Typické volanie funkcie `newsom` však vynecháva väčšinu parametrov:

```
net=newsom(p,[3, 4], 'gridtop');
```

Úloha: Natrénujte Kohonenovu mapu s pravouhlou topológiou pre množinu vektorov, ktoré vracia funkcia ksvi.mff.cuni.cz/~mraz/nn/dataset2.m. Vyskúšajte rôzne počty neurónov a rôzne topológie siete.