Dense_Associative_Memory_training Krotov - Hopfield - Fórmulas y comentarios

Thursday, September 28, 2023 3:15 PM

M = (784,60000) (M: **train data**)
(pixels in rows; train data in each column)

Pixels↓/data →	1	2	....	60.000
p				
i				
x				
e				
l				
s				

MT = (784,10000) (M: **test data**)

Lab = (10,60000) - labels from train ds
The corresponding label is indicated with a value of 1 in the corresponding classification neuron (rows).

		Samples			
digit	Class. Neuron ↓/ samples →	1	2	...	60.000
0	-1	-1	1		-1
1	-1	-1	-1		1
2	-1	1	-1		-1
3	1	-1	-1		-1
4	-1	-1	-1		-1
...	...	...	...		...
9	-1	-1	-1		-1

LabT = (10,10000) - labels from test ds

mini_code_tests.py

```
#%%-----------------------------------------------------------------
Nc1=3   # number of classification neurons
Num1 = 4  # Size of training minibatch
Aux1 = -np.ones((Nc1,Num1*Nc1))  #
for d1 in range(Nc1):
    print(d1,d1*Num1,(d1+1)*Num1)
    aux1[d1,d1*Num1:(d1+1)*Num1]=1.
    print(aux1)


[[ 1.  1.  1.  1. -1. -1. -1. -1. -1. -1. -1. -1.]
 [-1. -1. -1. -1.  1.  1.  1.  1. -1. -1. -1. -1.]
| [-1. -1. -1. -1. -1. -1. -1. -1.  1.  1.  1.  1.]]

En amarillo: minibatch
Cada minibatch es inicializado con solo una neurona en "on" state (+1)
Un minibatch es una columna. En este ej. tengo 3 "0"s, 3 "1"s, 3 "2"s
```

```
N=784  # number o pixels (= number of visible neurons)
Nc=10  # number of classification neurons
Ns=60000 # size of train dataset (number of samples)
NsT=10000 # size of test dataset (number of samples)

Kx=10               # Number of memories per row on the weights plot
Ky=10               # Number of memories per column on the weights plot
K=Kx*Ky             # Number of memories
n=20                # Power of the interaction vertex in the DAM energy function | power of x^n in F(x)?
m=30                # Power of the loss function
eps0=4.0e-2         # Initial learning rate
f=0.998             # Damping parameter for the learning rate
p=0.6               # Momentum
# Nep=300             # Number of epochs
Nep=1             # Number of epochs
Temp_in=540.        # Initial temperature
Temp_f=540.         # Final temperature
thresh_pret=200     # Length of the temperature ramp
Num=1000            # Size of training minibatch
NumT=5000           # Size of test minibatch
mu=-0.3             # Weights initialization mean
sigma=0.3           # Weights initialization std
prec=1.0e-30        # Precision of weight update


# KS = (100,794) | 794 = 784 (pixels) + 10 (classif neuron states - x_alpha)
KS  # weights initialization (including the 10 classificacion neurons).
VKS # auxiliary matrix variable for weight update calculation
```

$$E = -\frac{1}{2}\sum_{i,j=1}^{N}\sigma_i T_{ij}\sigma_j, \quad T_{ij} = \sum_{\mu=1}^{K}\xi_i^\mu \xi_j^\mu, \qquad (1)$$

Aquí está implícita una realimentación
(de x_alpha a c_alpha): es decir, se realimentan
Los estados de las neuronas

$$E = -\sum_{\mu=1}^{K} F(\xi_i^\mu \sigma_i) \qquad (2)$$

rectified polynomial energy function

$$F(x) = \begin{cases} x^n, & x \ge 0 \\ 0, & x < 0 \end{cases} \qquad (3)$$

$$\sigma_i^{(t+1)} = Sign\Big[\sum_{\mu=1}^{K}\Big(F\Big(\xi_i^\mu + \sum_{j\neq i}\xi_j^\mu \sigma_j^{(t)}\Big) - F\Big(-\xi_i^\mu + \sum_{j\neq i}\xi_j^\mu \sigma_j^{(t)}\Big)\Big)\Big], \qquad (4)$$

KSvv**n   Ksuu**n

$$c_\alpha = g\Big[\beta\sum_{\mu=1}^{K}\Big(F\Big(-\xi_\alpha^\mu x_\alpha + \sum_{\gamma\neq\alpha}\xi_\gamma^\mu x_\gamma + \sum_{i=1}^{N}\xi_i^\mu v_i\Big) - F\Big(\xi_\alpha^\mu x_\alpha + \sum_{\gamma\neq\alpha}\xi_\gamma^\mu x_\gamma + \sum_{i=1}^{N}\xi_i^\mu v_i\Big)\Big)\Big], \quad (9)$$

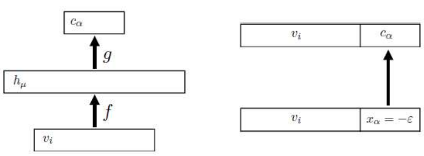

Figure 3:

Creo que este gráfico solo muestra que una neurona de clasificación
Se le hace un flip, para luego evaluar el cambio de energía y actualizar la red

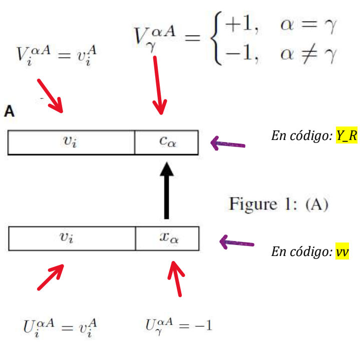

Figure 1: (A)

```
u = np.concatenate((v, -np.ones((Nc,Num))),axis=0)
```

$$c_\alpha \approx g\Big[\beta\sum_{\mu=1}^{K} F'\Big(\sum_{i=1}^{N}\xi_i^\mu v_i\Big)\,(-2\xi_\alpha^\mu x_\alpha)\Big] = g\Big[\sum_{\mu=1}^{K}\xi_\alpha^\mu\, F'(\xi_i^\mu v_i)\Big] = g\Big[\sum_{\mu=1}^{K}\xi_\alpha^\mu\, f(\xi_i^\mu v_i)\Big], \quad (10)$$

effective temperature $\beta = 1/T^n$

$$\varepsilon(t) = \varepsilon_0 f^t, \qquad f = 0.998, \qquad (12)$$

$$V_I^\mu(t) = pV_I^\mu(t-1) - \partial_{\xi_I^\mu} C$$
$$\xi_I^\mu(t) = \xi_I^\mu(t-1) + \varepsilon \frac{V_I^\mu(t)}{\max\limits_J |V_J^\mu(t)|}, \qquad (13)$$

No contiene solo los pesos sino también los estados de las neuronas

$$C = \sum_{\substack{\text{training}\\ \text{examples}}} \sum_{\alpha=1}^{N_c} (c_\alpha - t_\alpha)^{2m}, \qquad (14)$$

$$\partial_{\xi_I^\mu} C = (2m\beta n) \sum_{A=1}^{M} \sum_{\alpha=1}^{N_c} (c_\alpha^A - t_\alpha^A)^{2m-1} \left[1 - (c_\alpha^A)^2\right] \left[F_{n-1}(\xi_J^\mu V_J^{\alpha A}) V_I^{\alpha A} - F_{n-1}(\xi_J^\mu U_J^{\alpha A}) U_I^{\alpha A}\right] \qquad (17)$$

$$U_i^{\alpha A} = v_i^A \qquad V_i^{\alpha A} = v_i^A$$
$$U_\gamma^{\alpha A} = -1 \qquad V_\gamma^{\alpha A} = \begin{cases} +1, & \alpha = \gamma \\ -1, & \alpha \neq \gamma \end{cases} \qquad (15)$$

Gamma = 0…9

Vi (vv) = [vi … aux]

$$c_\alpha^A = g\left[\beta\Big(\sum_{\mu=1}^{K} F_n(\xi_J^\mu V_J^{\alpha A}) - F_n(\xi_J^\mu U_J^{\alpha A})\Big)\right], \qquad (16)$$

$$c_\alpha = g\left[\beta \sum_{\mu=1}^{K} \left(F\Big(-\xi_\alpha^\mu x_\alpha + \sum_{\gamma\neq\alpha} \xi_\gamma^\mu x_\gamma + \sum_{i=1}^{N} \xi_i^\mu v_i\Big) - F\Big(\xi_\alpha^\mu x_\alpha + \sum_{\gamma\neq\alpha} \xi_\gamma^\mu x_\gamma + \sum_{i=1}^{N} \xi_i^\mu v_i\Big)\right)\right], \quad (9)$$

$$\partial_{\xi_I^\mu} C = (2m\beta n) \sum_{A=1}^{M} \sum_{\alpha=1}^{N_c} (c_\alpha^A - t_\alpha^A)^{2m-1} \left[1 - (c_\alpha^A)^2\right] \left[F_{n-1}(\xi_J^\mu V_J^{\alpha A}) V_I^{\alpha A} - F_{n-1}(\xi_J^\mu U_J^{\alpha A}) U_I^{\alpha A}\right] \qquad (17)$$