System Architecture

* 全域變數

#define SUCCESS\_INIT 0

#define ERR\_INIT 1

#define SUCCESS\_DINIT 0

#define ERR\_DINIT 1

#define SUCCESS\_INIT\_WEIGHTS 0

#define ERR\_INIT\_WEIGHTS 1

#define SUCCESS\_UPDATE\_WEIGHTS 0

#define SUCCESS\_CREATE\_ARCHITECTURE 0

#define ERR\_CREATE\_ARCHITECTURE 1

typedef struct layer\_t{

    int num\_neu;

    struct neuron\_t \*neu;

} layer;

layer create\_layer(int num\_neurons);

layer \*lay = NULL;

typedef struct neuron\_t{

    float actv, \*out\_weights, bias, z;

    float dactv, \*dw, dbias, dz;

} neuron;

neuron create\_neuron(int num\_out\_weights);

int num\_layers;

int \*num\_neurons;

float alpha;

float \*cost;

float full\_cost;

float \*\*input;

float \*\*desired\_outputs;

float error=0;

float SUM=0;

int num\_training\_ex;

int n=1;

int it;

FILE \*fptr;

* int init( )

判斷函式create\_architecture( )是否成功。若成功則輸出”Neural Network Created Successfully”，回傳SUCCESS\_INIT (0)；失敗則輸出” Error in creating architecture”，回傳ERR\_INIT (1)。

* int create\_architecture( )

依照使用者輸入的layer數(num\_layers)與電腦要空間，再呼叫函式create\_layer(num\_neurons[i])以及create\_neuron(num\_neurons[i+1])建立各個layer與neuron，然後將輸入的neuron儲存於lay.num\_neu中。最後呼叫函式initialize\_weights( )，初始化類神經網路的權重。若成功則回傳SUCCESS\_CREATE\_ARCHITECTURE (0)；若失敗則輸出”Error Initilizing weights”，並回傳ERR\_CREATE\_ARCHITECTURE (1)。

* layer create\_layer(int )

依照layer中neuron的數量跟電腦要空間(lay.neu)。

* neuron create\_neuron(int )
* int initialize\_weights( )

使用隨機函式rand( )，根據layer以及neuron的數量產生相應數量的權重

(lay.neu.out\_weights)，且權重皆小於一。