

**Instituto Politécnico Nacional**

**Escuela Superior de Cómputo**

*Asignatura*:

Neural Networks

*Grupo*: 3CM2

Práctica EXTRA. **MLP ejercicio A**

*Alumno*:

Garcia Garcia Rafael

*Profesor*: Moreno Armendáriz Marco Antonio

**Experimentos:**

**Primera parte :** *Capturar la ecuación y generar 100 puntos en el rango indicado*

Para esta parte de la practica se tuvo que crear una función llamada *generar\_polinomio\_y\_target* que generara 2 archivos: “Polinomio\_Entrada.txt” y “Polinomio\_Target.txt”. Cada uno de estos archivos nos servirán para el entrenamiento de nuestro MLP. A continuación, les mostraré la salida de esta función que se anexo a nuestro MLP y los vectores de entrada y target para comprobarlo.

Experimento 1:

|  |  |  |
| --- | --- | --- |
| >> generar\_polinomio\_y\_target  Introduzca el rango  Ejemplos:  [2.7, 7.5]  >>[2.7, 7.5]  Introduzca la ecuación de G(p):  Ejemplos:  ( sin(p)+sin((10/3).\*p) )  (sin(p)+sin((10/3).\*p)+log(p)-(0.84.\*p)+3)  >>( sin(p)+sin((10/3).\*p) )  Numero de puntos generados en p  100  Numero de puntos generados en t  100 | Desplegando p  2.7000  2.7485  2.7970  2.8455  2.8939  2.9424  2.9909  3.0394  3.0879  3.1364  3.1848  3.2333  3.2818  3.3303  3.3788  3.4273  3.4758  3.5242  3.5727  3.6212  3.6697  3.7182  3.7667  3.8152  3.8636  3.9121  3.9606  4.0091  4.0576  4.1061  4.1545  4.2030  4.2515  4.3000  4.3485  4.3970  4.4455  4.4939  4.5424  4.5909  4.6394  4.6879  4.7364  4.7848  4.8333  4.8818  4.9303  4.9788  5.0273  5.0758  5.1242  5.1727  5.2212  5.2697  5.3182  5.3667  5.4152  5.4636  5.5121  5.5606  5.6091  5.6576  5.7061  5.7545  5.8030  5.8515  5.9000  5.9485  5.9970  6.0455  6.0939  6.1424  6.1909  6.2394  6.2879  6.3364  6.3848  6.4333  6.4818  6.5303  6.5788  6.6273  6.6758  6.7242  6.7727  6.8212  6.8697  6.9182  6.9667  7.0152  7.0636  7.1121  7.1606  7.2091  7.2576  7.3061  7.3545  7.4030  7.4515  7.5000 | Desplegando t  0.8395  0.6432  0.4392  0.2318  0.0253  -0.1761  -0.3682  -0.5472  -0.7094  -0.8520  -0.9721  -1.0680  -1.1382  -1.1820  -1.1995  -1.1913  -1.1588  -1.1038  -1.0289  -0.9371  -0.8318  -0.7167  -0.5960  -0.4735  -0.3536  -0.2400  -0.1368  -0.0473  0.0254  0.0786  0.1102  0.1189  0.1037  0.0647  0.0022  -0.0825  -0.1876  -0.3107  -0.4489  -0.5987  -0.7564  -0.9181  -1.0795  -1.2366  -1.3850  -1.5209  -1.6405  -1.7405  -1.8181  -1.8708  -1.8969  -1.8952  -1.8655  -1.8078  -1.7231  -1.6131  -1.4798  -1.3260  -1.1550  -0.9704  -0.7762  -0.5766  -0.3758  -0.1782  0.0121  0.1912  0.3553  0.5013  0.6265  0.7286  0.8061  0.8582  0.8846  0.8858  0.8628  0.8174  0.7520  0.6695  0.5730  0.4662  0.3531  0.2377  0.1242  0.0164  -0.0817  -0.1664  -0.2347  -0.2838  -0.3114  -0.3158  -0.2963  -0.2523  -0.1842  -0.0931  0.0193  0.1509  0.2988  0.4597  0.6300  0.8056 |

**Salida de terminal 1.** G(p)=( sin(p)+sin((10/3).\*p) )

**Segunda parte:** Realizar el adecuado aprendizaje de las ecuaciones, llegando al menos a un error de prueba de 1x10-3

Para la primera función: G(p)=( sin(p)+sin((10/3).\*p) )

La siguiente es la tabla de experimentación

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Intento** | **Early Stopping** | **Arquitectura** | **# de épocas** | **Alpha** | **Error de entrenamiento** | **Error**  **de validación** | **Error de prueba** |
| 1 | NO | V1 = [1,7,7,1]  V2 = [2,2,1] | 12,000 | 0.009 | 0.5729 | 0.7058 | 1.2751 |
| 2 | NO | V1 = [1,10,7,1]  V2 = [3,3,1] | 12,000 | 0.009 | 0.6756 | 0.6635 | 0.6682 |
| 3 | NO | V1 = [1,8,5,1]  V2 = [2, 2, 1] | 71,999 | 0.009 | 3.283x10-5 | 5.359x10-5 | 3.1321x10-5 |
| **Tabla 1.** Experimentos realizados para G(p)=( sin(p)+sin((10/3).\*p) ) | | | | | | | |

**Salida de la terminal:**

|  |
| --- |
| Error: 0.000033 de la epoca 71997  Error: 0.000033 de la epoca 71998  Error: 0.000033 de la epoca 71999  Prueba de generalización superada con un error de:  3.1321e-05  Error final  3.2833e-05  Valores de pesos finales:  W1  -0.8867  -1.4270  -1.0759  -2.3908  -1.4436  -3.4046  1.2302  0.8592  b\_1  6.1107  5.9155  0.7134  6.2323  10.4573  12.6893  -7.1565  -3.2358  W\_2  3.0478 3.0775 -0.9754 1.0321 4.3466 8.8322 -4.7283 -3.2542  0.1924 0.1843 0.7070 6.3966 0.1628 -3.3124 1.5572 -1.3339  -3.5488 -3.1608 -0.4512 -1.4774 -5.8096 -3.1450 4.5512 1.7704  -3.3602 -2.6654 0.4485 0.5847 -5.0714 -1.8381 2.8159 1.0776  -2.1466 -3.1079 0.0459 -1.3366 -2.3715 -7.8788 4.6433 3.8487  b\_2  -1.8909  1.5317  1.4095  -0.8535  1.2602 |
| **Tabla 2.** Salida de la terminal del programa para G(p)=( sin(p)+sin((10/3).\*p) ) |

*Gráficas:*

|  |
| --- |
| **Gráfica del error** |
|  |
| **Graficas de W y b** |
|  |
|  |
|  |
|  |
|  |
|  |
| Salida de la Red Vs salida del conjunto de prueba |
|  |
| **Tabla 3.** Gráficas de la función: G(p)=( sin(p)+sin((10/3).\*p) ) |

**Discusión**

Debido a que el MLP se había implementado de una excelente manera en la práctica 5, fue muy fácil agregar la parte de la ecuación para generar nuevas funciones de las cuales aprenda. También no fue muy difícil jugar con la red neuronal para obtener los resultados, debido a que tener muchas neuronas y 2 capas personalizables da muy buenos resultados para funciones seno.

Como era de esperarse tuvimos buenos resultados y nuestro MLP funciono de manera eficiente. Algo interesante es de que en la segunda función copie la misma configuración que en la primera función pero al ser más compleja esta tardo mucha más épocas para converger, además de que tuvo un error ligeramente mas grande pero ambos errores dentro de los objetivos esperados en la clase

**Conclusiones:**

Haber desarrollado esta práctica me fue de gran utilidad para agregar funcionalidades extras a mi MLP (Como la introducción de la ecuación y rango) y también para verificar que puede funcionar para diversas funciones el mismo perceptrón, dependiendo de la complejidad de la función tardará más o menos iteraciones en converger y creo que esta práctica aporto mas conocimientos a mi formación durante la asignatura y reafirmo los que ya tenía.

Fue un excelente curso me lleve muchos conocimientos y su compromiso profesor me ayudo a adquirir los conocimientos esperados de manera eficiente a pesar de la pandemia. :D

# Bibliografía

Hagan, M. T. (s.f.). Neural Network Design. En M. T. Hagan, *Neural Network Design* (pág. 1010).