Sistemas de Inteligencia Artificial

TP1: Redes Neuronales

Marcos Abelenda, 55086
Santiago Manganaro Bello, 56239
Matias Perazzo, 55024
Agustín Ignacio Vázquez, 55354

Introducción

Dos etapas: aprendizaje y testeo. Set de datos dividido.

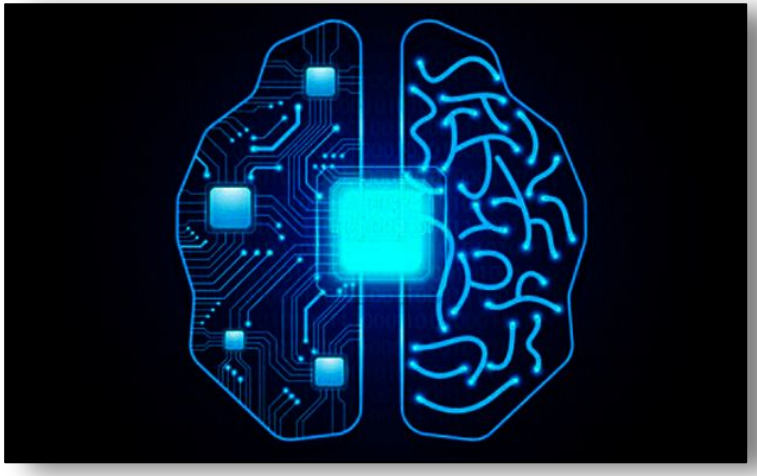
Se tomaron distintas métricas de las arquitecturas propuestas:

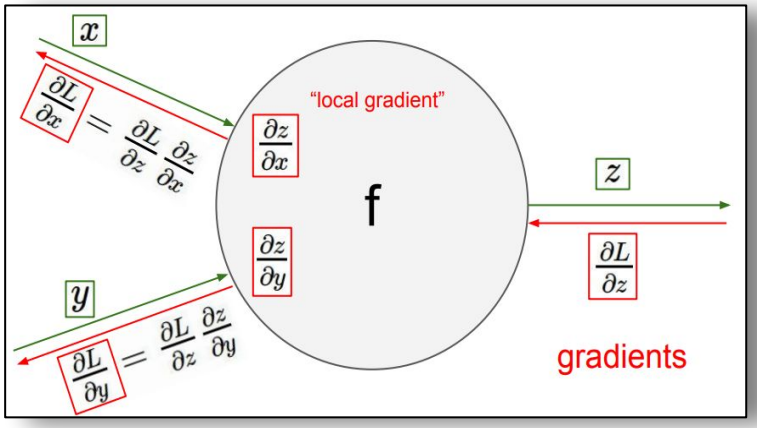
- Errores cuadráticos medios de aprendizaje y de testeo.
- Cantidad de aciertos de la red.
- Rapidez de convergencia.

Técnicas de mejoras para las redes posibles.

Verificación empírica de las mejoras

Back propagation incremental.



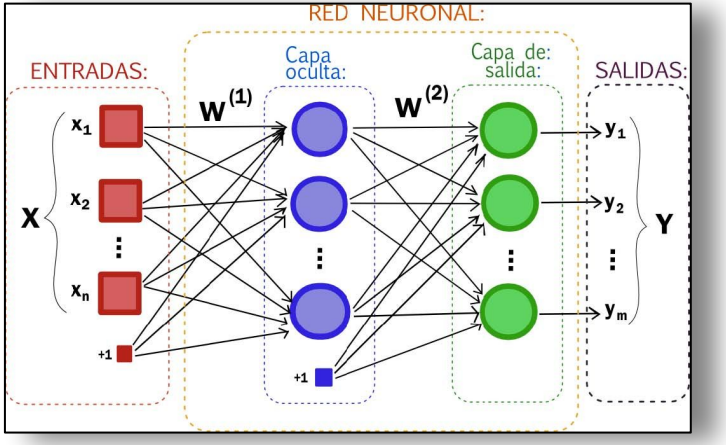


Arquitectura elegida

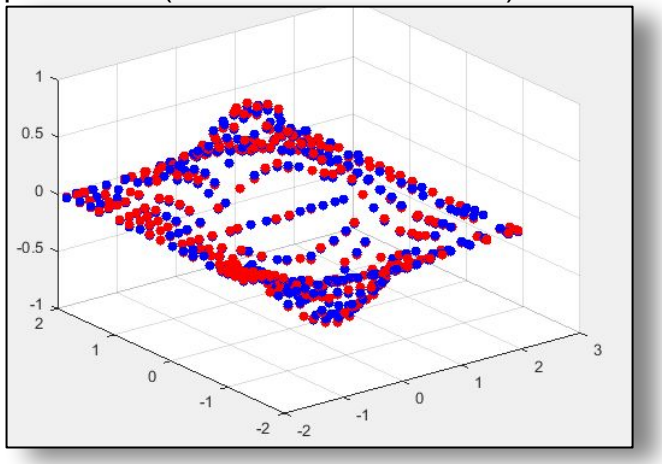
- Tiempo
- Precisión
- Generalidad

Menor error cuadrático medio al finalizar 3000 épocas.
Capacidad de generalización por sobre la de aprendizaje.
Para una semilla determinada, no para la mayor cantidad de semillas.

Cantidad de entradas: 2
Cantidad de neuronas primer capa: 15
Cantidad de neuronas segunda capa: 10
Cantidad de salidas: 1



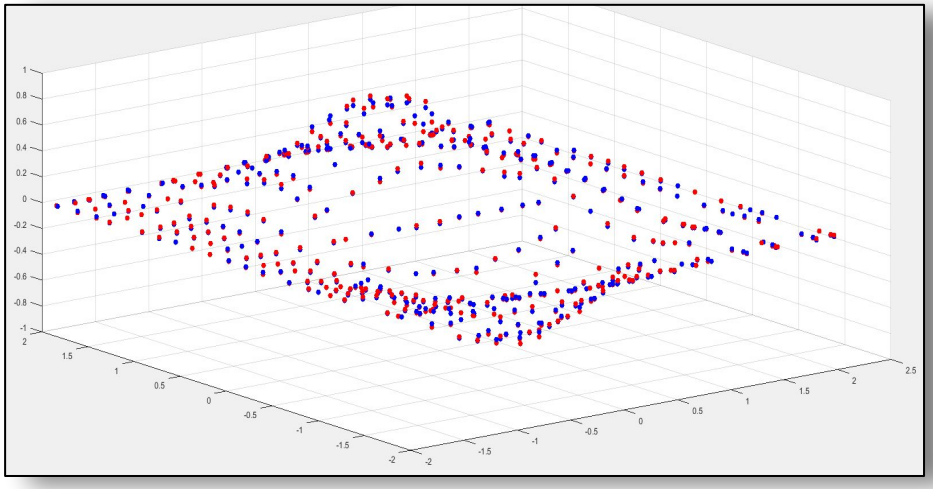
Nota: los puntos rojos representan la solución exacta y los azules la aproximada (la salida de nuestra red).



Arquitectura elegida

- 1,14E-04 de error cuadrático medio de entrenamiento.
- 1,19E-04 de error cuadrático medio de testeo.
- 88% de aciertos en el aprendizaje.
- 86,52% de aciertos para la generalización.
- 286,196 segundos en hacer 3000 épocas y llegar a estos valores.
- El epsilon para los aciertos es de 2E-02.

Nota: los puntos rojos representan la solución exacta y los azules la aproximada (la salida de nuestra red).



W1

-0.1305 1.4477 -0.7623
0.6912 0.2742 0.3718
-0.8544 0.1365 0.9206
0.0562 -0.5950 0.3419
-0.2874 -0.6941 0.6423
-0.5789 1.0295 -0.0594
-0.0273 0.7182 0.2675
-0.0608 -0.8459 1.2523
0.6214 1.0738 -0.4536
0.6289 0.7455 0.5846
-0.2016 0.9440 0.2984
0.3689 -0.5325 0.5092
-0.0669 -0.4387 0.4126
0.7724 0.1557 0.7883
-0.7863 0.9386 -1.0703

W2

0.6115 0.1890 0.5389 0.135 2 0.7856 0.1332
1.1031 0.6104 0.7922 0.7976 0.4923 0.8819
0.3463 0.0070 0.2562 0.1135 1.0432 0.6117
0.1069 0.6130 0.7894 0.4447 0.5755 -0.0938
0.0115 0.6453 0.5477 0.8325 0.6466 0.6718
-0.0147 0.0680 0.8109 0.9970 0.1522 0.4477
0.5799 0.4002 0.0099 0.6073 0.9841 0.5662
-0.1849 0.5310 0.0447 0.9263 0.5395 1.0542
0.1764 0.0746 0.8013 0.3349 0.5661 0.3672
0.3928 -0.2420 0.4974 0.2688 0.3732 0.8291
0.5475 0.7310 0.4873 0.4682 0.1258 0.7477
-0.1930 0.6612 0.4080 0.7319 0.5254 0.0360
0.7472 0.6697 0.5557 0.2305 0.2901 0.3109
0.8098 0.5378 0.2904 0.9520 0.3394 0.6806
0.9938 0.1453 0.2736 0.6483 0.0034 0.8271
0.1852 0.0832 0.7554 0.5999 0.0428 0.3973
-0.0054 -0.1354 0.9943 -0.0009 0.8174 0.3399
1.0049 0.0034 0.0744 -0.4659 0.4523 0.4235
0.3986 0.2009 -1.0133 -0.2183 0.1400 0.6803
-0.0392 0.6705 0.6703 0.1752 0.2775 -0.1225
-0.0896 0.7007 0.7935 1.0662
0.3743 0.7270 0.4102 -0.5601
0.4498 0.3220 0.5578 0.2174
0.4551 -0.1014 0.3049 0.8175
0.5552 0.4179 0.9052 0.9196
0.2856 0.6819 0.2665 1.1417
1.0131 0.2618 0.0265 0.3624
0.4016 0.7962 0.5599 0.9237
0.1683 0.1026 0.4243 0.3848
0.4881 0.2304 1.0648 0.3113

W3

0.0029 0.5848
-0.7740 0.2333
0.0622 0.2172
0.6098 -0.1091
-0.5856 -0.9480
0.6068

Optimizaciones

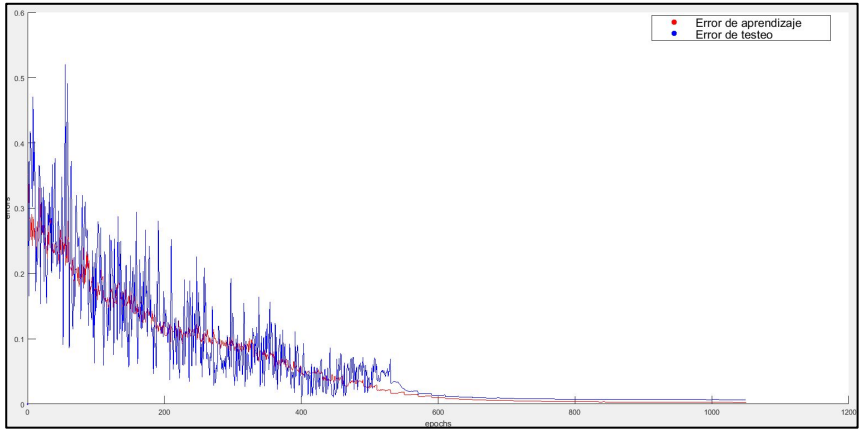
Saturación de la red

- Normalización

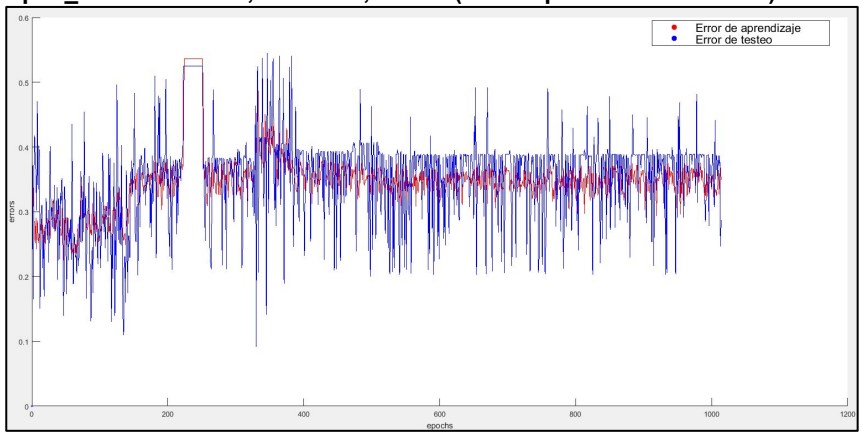
Mínimos locales

- Shuffling del set de entrenamiento para mínimos locales no profundos
- Eta adaptativo para mínimos locales profundos. Se analizan los últimos n pasos y su tendencia.

Red 2,15,10,1 con: eta_check_steps 10, eta_increase_value 0.001, eta_decrease_factor 0.95, training_size 300, eta 0.5, shuffle_flag 1, alpha_momentum 0.5, seed “D”, tanh 1



Red 2,15,10,1 con: training_size 300, eta 0.5, shuffle_flag 1, alpha_momentum 0.5, seed “D”, tanh 1 (eta adaptativo desactivado)



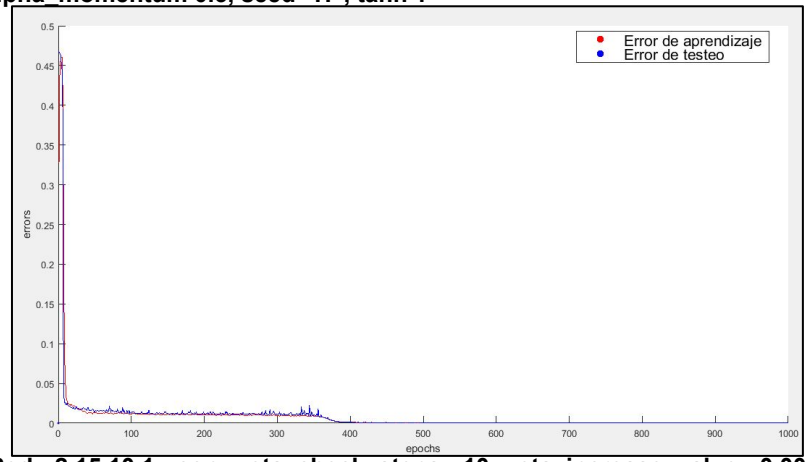
Optimizaciones

Convergencia

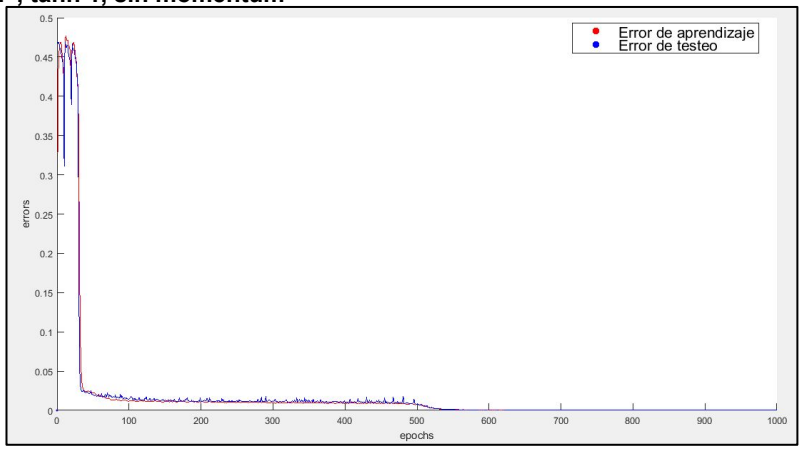
- Momentum y eta adaptativo. Objetivo: acelerar convergencia. El primero, suma una proporción de la variación de los pesos anteriores. El segundo, incrementa el eta cuando es necesario. Ambos hacen que nos movamos aún más en la superficie en la dirección actual.

- Cota mínima de error para finalizar el entrenamiento de la red cuando se llega a un error aceptable. Se utiliza para comparar tiempos (comparar arquitecturas) y para evitar sobre aprendizaje.

Red 2,15,10,1 con: eta_check_steps 10, eta_increase_value 0.001, eta_decrease_factor 0.95, training_size 300, eta 0.01, shuffle_flag 1, alpha_momentum 0.5, seed "H", tanh 1



Red 2,15,10,1 con: eta_check_steps 10, eta_increase_value 0.001, eta_decrease_factor 0.95, training_size 300, eta 0.01, shuffle_flag 1, seed "H", tanh 1, sin momentum



Optimizaciones

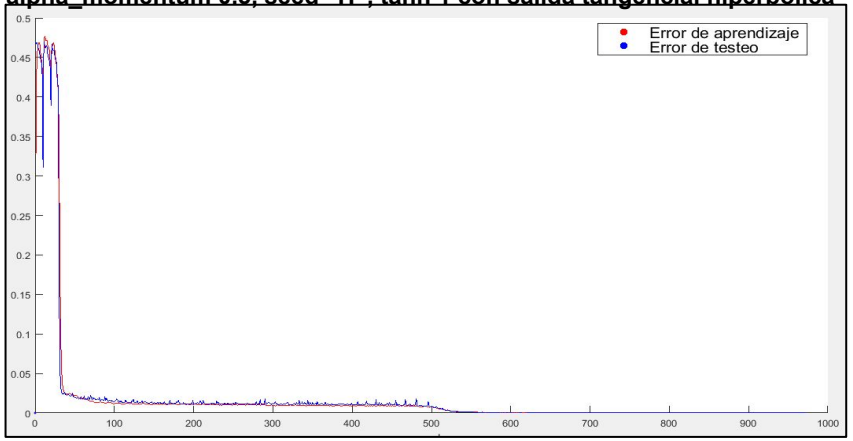
Capa de salida

- Dado que la salida del terreno a aproximar se encuentra entre -1 y 1 se utilizó una función de activación en la capa de salida en vez de una tangencial.

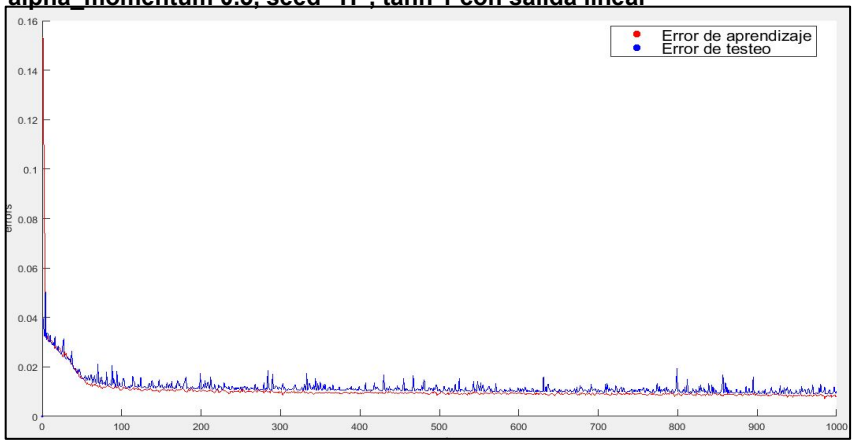
Solución aproximada

- Semillas óptimas.
- Peso óptimo. Se almacenan los pesos de la red con menor error cuadrático.

Red 2,15,10,1 con: eta_check_steps 10, eta_increase_value 0.001, eta_decrease_factor 0.95, training_size 300, eta 0.01, shuffle_flag 1, alpha_momentum 0.5, seed “H”, tanh 1 con salida tangencial hiperbólica



Red 2,15,10,1 con: eta_check_steps 10, eta_increase_value 0.001, eta_decrease_factor 0.95, training_size 300, eta 0.01, shuffle_flag 1, alpha_momentum 0.5, seed “H”, tanh 1 con salida lineal

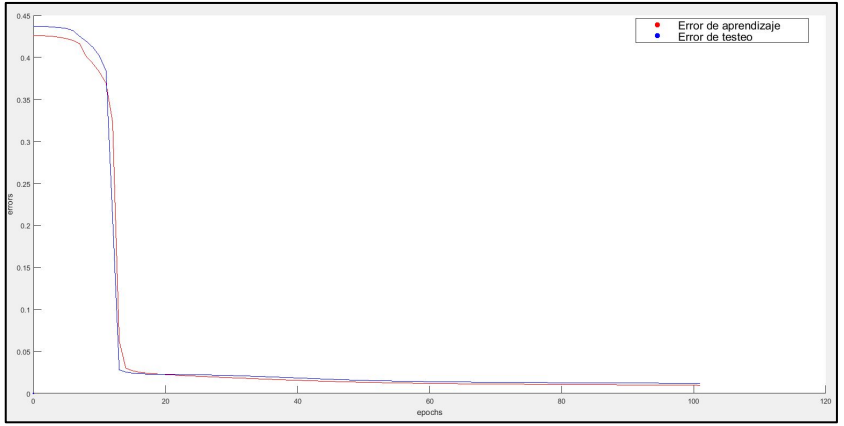


Observaciones

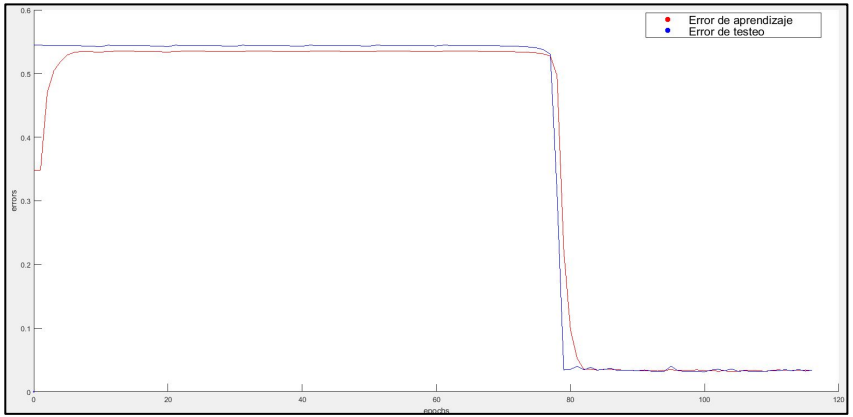
Obtenidas a partir de más de 50 casos de prueba.

- Reducción del tamaño de entrenamiento implica mayor error.
- Función tangencial vs exponencial.
- Momentum.
- Diferentes semillas para distintas redes.
- Error de aprendizaje no implica error de testeo; y viceversa.
- Shuffle, precisión > velocidad. No siempre conviene.
- Eta adaptativo. 35% de mejora aprendizaje decreciente. 25% de mejora testeo decreciente. (1000 épocas)

Red 2,15,10,1 con: eta_check_steps 10, eta_increase_value 0.001, eta_decrease_factor 0.95, training_size 300, eta 0.01, shuffle_flag 1, alpha_momentum 0.5, seed “D”,, tanh



Red 2,15,10,1 con: eta_check_steps 10, eta_increase_value 0.001, eta_decrease_factor 0.95, training_size 300, eta 0.01, shuffle_flag 1, alpha_momentum 0.5, seed “D”, tanh 0 (exponencial)



Conclusiones

- No se puede garantizar que la red elegida sea la mejor.
- La forma en la que un sistema busca la solución (como recorre la superficie) se ve afectada por la semilla en cuestión. No se puede asegurar que una arquitectura es mejor que otra por este mismo motivo.
- Tanto eta adaptativo como momentum logran una mayor rapidez de convergencia.
- Para nuestro problema la tangente hiperbólica es mejor que la exponencial.

- El shuffle demostró influir positivamente en la convergencia en largo plazo en una mayor medida que en la de corto plazo.
- Si bien un sistema puede aprender mejor que otro, esto no implica que el mismo tenga una mayor capacidad de generalización.
- El tamaño de los patrones de entrada para el entrenamiento incide directamente con la capacidad de generalización de la red.
- A la hora de elegir una red, el *trade-off* siempre va a existir; es trabajo del diseñador poder determinar cuales son los requisitos y en qué orden de prioridad se encuentran para poder determinar el mejor resultado.