Alvaro Zarabanda (20161020507)
Nayib Moreno (20152020401)

TALLER DE REDES NEURONALES

Para la realización del presente taller, se tuvieron en cuenta los nombres de los integrantes del grupo, cada letra, incluido el espacio se transformo a binario utilizando código ASCII, posterior a ello se aplicó la operación XOR a cada uno de estas letras utilizando una cadena binaria aleatoria, así obteniendo la siguiente información:

	ASCII	**CADENA**	**XOR**
A	0100 0001	1100 1110	**1000 1111**
L	0100 1100	1100 1110	**1000 0010**
V	0101 0110	1100 1110	**1001 1000**
A	0100 0001	1100 1110	**1000 1111**
R	0101 0010	1100 1110	**1001 1100**
O	0100 1111	1100 1110	**1000 0001**
	0010 0000	1100 1110	**1110 1110**
N	0100 1110	1100 1110	**1000 0000**
A	0100 0001	1100 1110	**1000 1111**
Y	0101 1001	1100 1110	**1001 0111**
I	0100 1001	1100 1110	**1000 0111**
B	0100 0010	1100 1110	**1000 1100**

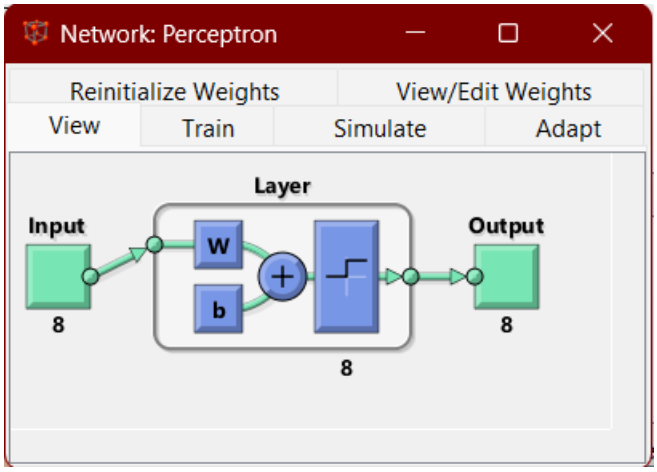
Con dicha información, se plantearán 3 esquemas de redes neuronales, que como entradas tendrán el resultado obtenido de la operación XOR y como salida se desea obtener la misma cadena de entrada. Dicho esto, se plantea la siguiente tabla:

ENTRADAS								**SALIDAS**							
Pos1	**Pos2**	**Pos3**	**Pos4**	**Pos5**	**Pos6**	**Pos7**	**Pos8**	**Pos1**	**Pos2**	**Pos3**	**Pos4**	**Pos5**	**Pos6**	**Pos7**	**Pos8**
1	0	0	0	1	1	1	1	0	1	0	0	0	0	0	1
1	0	0	0	0	0	1	0	0	1	0	0	1	1	0	0
1	0	0	1	1	0	0	0	0	1	0	1	0	1	1	0
1	0	0	0	1	1	1	1	0	1	0	0	0	0	0	1
1	0	0	1	1	1	0	0	0	1	0	1	0	0	1	0
1	0	0	0	0	0	0	1	0	1	0	0	1	1	1	1
1	1	1	0	1	1	1	0	0	0	1	0	0	0	0	0
1	0	0	0	0	0	0	0	0	1	0	0	1	1	1	0
1	0	0	0	1	1	1	1	0	1	0	0	0	0	0	1
1	0	0	1	0	1	1	1	0	1	0	1	1	0	0	1
1	0	0	0	0	1	1	1	0	1	0	0	1	0	0	1
1	0	0	0	1	1	0	0	0	1	0	0	0	0	1	0

Alvaro Zarabanda (20161020507)
Nayib Moreno (20152020401)

ESQUEMA 1

En este caso se utilizaron perceptrones, con esto, obtenemos una única capa con 8 perceptrones, la función de activación es Hardlim:



Evaluando el desempeño de esta, podemos obtener muy buenos resultados para este caso, pues posterior al entrenamiento, los datos que predicen estos 8 perceptrones son correctos:

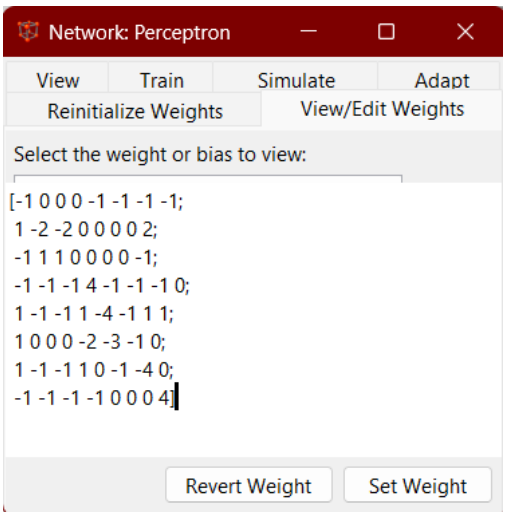
```
OUTPUT = [0 0 0 0 0 0 0 0 0 0 0 0;
          1 1 1 1 1 1 0 1 1 1 1 1;
          0 0 0 0 0 0 1 0 0 0 0 0;
          0 0 1 0 1 0 0 0 0 1 0 0;
          0 1 0 0 0 1 0 1 0 1 1 0;
          0 1 1 0 0 1 0 1 0 0 0 0;
          0 0 1 0 1 1 0 1 0 0 0 1;
          1 0 0 1 0 1 0 0 1 1 1 0];
```

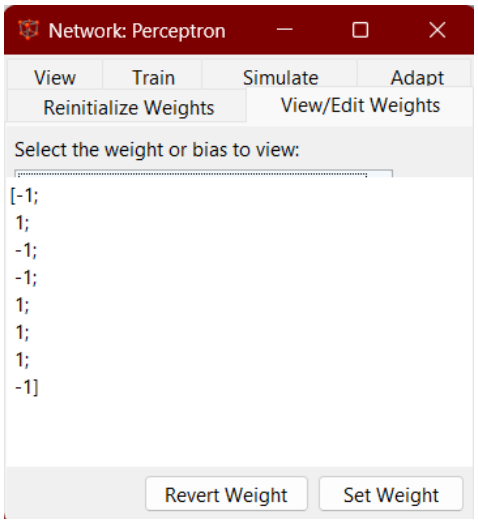
Data: Perceptron_outputs

Value

```
[0 0 0 0 0 0 0 0 0 0 0 0;
 1 1 1 1 1 1 0 1 1 1 1 1;
 0 0 0 0 0 0 1 0 0 0 0 0;
 0 0 1 0 1 0 0 0 0 1 0 0;
 0 1 0 0 0 1 0 1 0 1 1 0;
 0 1 1 0 0 1 0 1 0 0 0 0;
 0 0 1 0 1 1 0 1 0 0 0 1;
 1 0 0 1 0 1 0 0 1 1 1 0]
```

El entrenamiento finalizo con los siguientes pesos y bias:

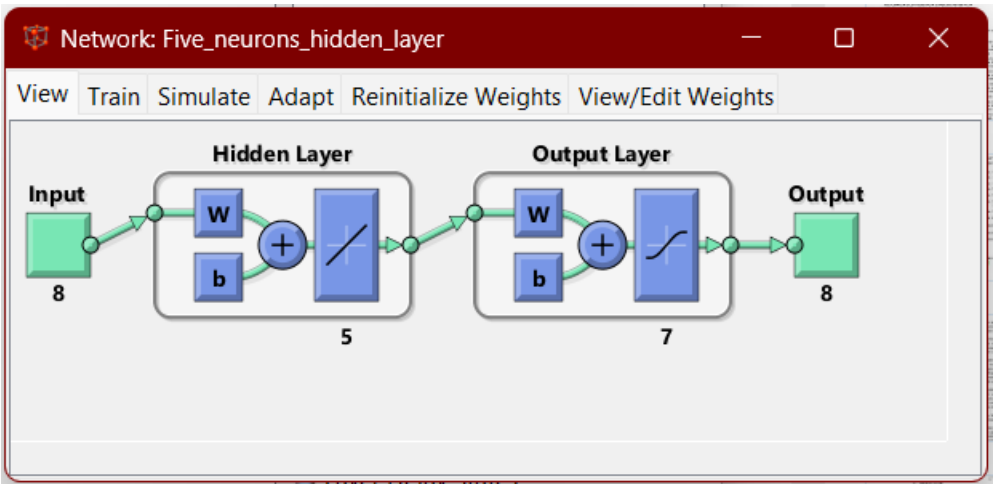




Alvaro Zarabanda (20161020507)
Nayib Moreno (20152020401)

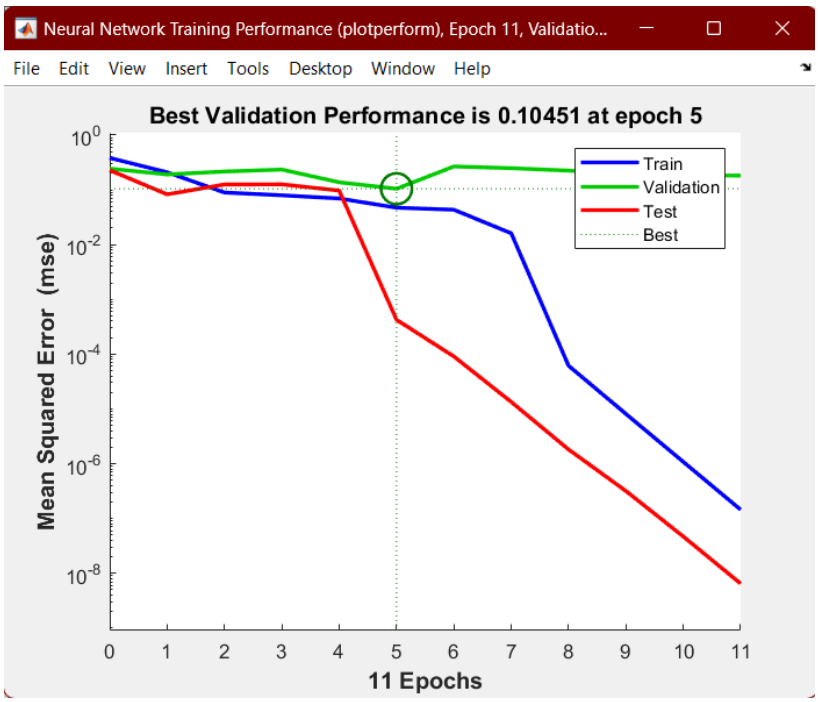
ESQUEMA 2

Ahora, se utilizará una red neuronal con una capa oculta de 5 neuronas



En donde para este caso la función de activación en la capa oculta fue Purelin y en la capa de salida Tansig

Posterior al entrenamiento, al cual se le dejaron todos los parámetros por defecto, se obtiene la gráfica con el desempeño de la red a lo largo de las épocas:



Se observa que en el conjunto de entrenamiento el error es un poco menor a comparación del conjunto de validación, en donde además ilustra cual fue la mejor época para ambos conjuntos, sin

Alvaro Zarabanda (20161020507)
Nayib Moreno (20152020401)

embargo, para los datos de prueba a medida que aumentaron más las épocas el error disminuyo, con esto, se mostrara la predicción hecha con todos los datos para continuar con el análisis:

```
OUTPUT = [0 0 0 0 0 0 0 0 0 0 0 0;
          1 1 1 1 1 1 0 1 1 1 1 1;
          0 0 0 0 0 0 1 0 0 0 0 0;
          0 0 1 0 1 0 0 0 0 1 0 0;
          0 1 0 0 0 1 0 1 0 1 1 0;
          0 1 1 0 0 1 0 1 0 0 0 0;
          0 0 1 0 1 1 0 1 0 0 0 1;
          1 0 0 1 0 1 0 0 1 1 1 0];
```

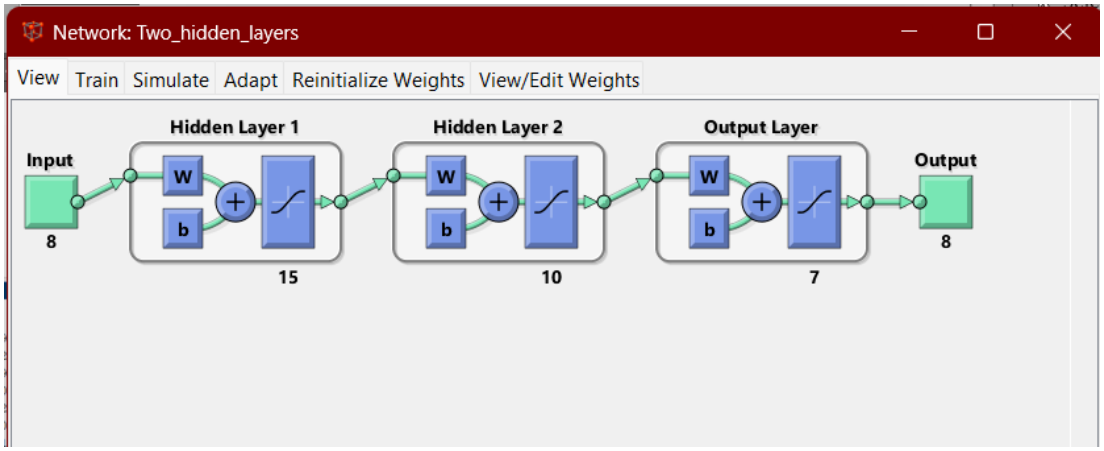
and Window

0	0	0	0	0	0	0	0	0	0	0	0
0.9711	0.9998	0.9930	0.9711	0.9995	0.9897	0.0055	0.9987	0.9711	1.0000	0.9999	0.9738
0.0420	0.0036	0.0051	0.0420	0.0266	0.0000	0.9983	0.0000	0.0420	0.1400	0.0006	0.0001
0.0000	0.0870	0.9715	0.0000	0.9988	0.0000	0.0257	0.0003	0.0000	0.9996	0.0001	0.0000
0.0282	1.0000	0.0141	0.0282	0.0064	0.9876	0.0000	0.9999	0.0282	1.0000	1.0000	0.0000
0.0002	1.0000	0.9830	0.0002	0.0222	0.9999	0.0024	1.0000	0.0002	0.9391	0.8778	0.0104
0.0001	0.0000	0.9820	0.0001	1.0000	0.0000	0.9917	0.0000	0.0001	0.0024	0.0000	0.9540
0.9975	0.1188	0.0000	0.9975	0.0131	0.9942	0.0102	0.0373	0.9975	1.0000	1.0000	0.0257

Ahora bien, se observa que no todos los valores son discretos, esto debido a la característica de la función de activación que se utilizó en la capa de salida, sin embargo, en la mayoría de casos se obtiene un valor muy cercano al deseado, con algunas excepciones.

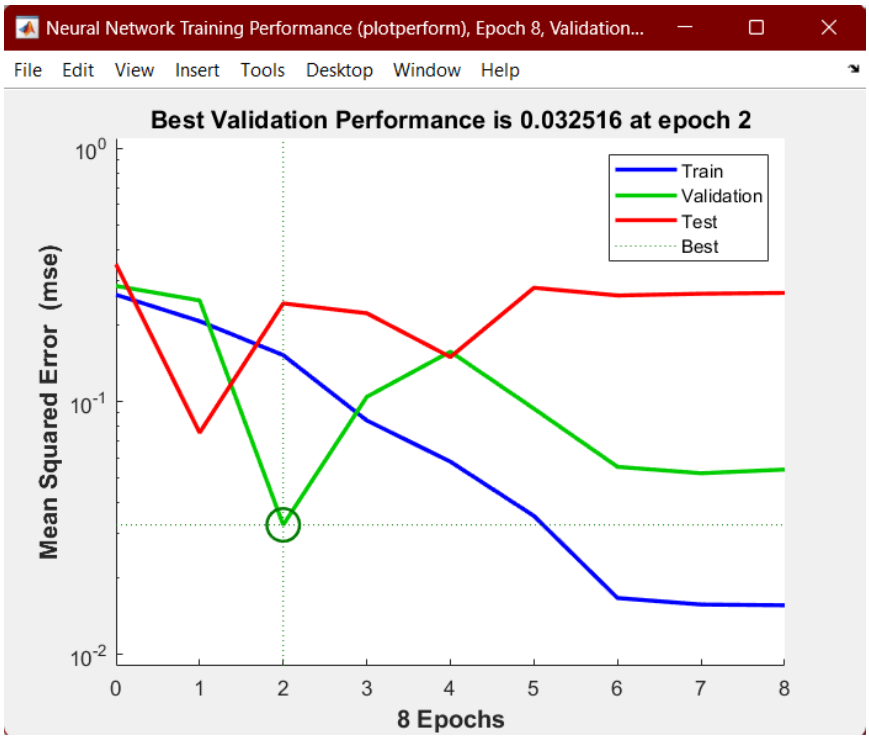
ESQUEMA 3:

Para este caso, se utilizarán 2 capas ocultas, con 15 y 10 neuronas respectivamente, todas con función de activación Tansig:



Posterior al entrenamiento se obtiene la siguiente grafica del desempeño:

Alvaro Zarabanda (20161020507)
Nayib Moreno (20152020401)



Similar al esquema 2, el error para los datos de prueba sigue siendo un poco menor que para los de validación, sin embargo, para los de prueba esta vez el error es mayor. Con este esquema se obtienen las siguientes predicciones:

```
OUTPUT = [0 0 0 0 0 0 0 0 0 0 0 0;
          1 1 1 1 1 1 0 1 1 1 1 1;
          0 0 0 0 0 0 1 0 0 0 0 0;
          0 0 1 0 1 0 0 0 0 1 0 0;
          0 1 0 0 0 1 0 1 0 1 1 0;
          0 1 1 0 0 1 0 1 0 0 0 0;
          0 0 1 0 1 1 0 1 0 0 0 1;
          1 0 0 1 0 1 0 0 1 1 1 0];
```

d Window

0	0	0	0	0	0	0	0	0	0	0	0
0.9931	0.9821	0.9980	0.9931	0.9989	0.9979	0.8122	0.9963	0.9931	0.9992	0.9804	0.9988
0.0063	0.0173	0.0018	0.0063	0.0029	0.0025	0.8391	0.0062	0.0063	0.0006	0.0138	0.0011
0.5263	0.1376	0.5073	0.5263	0.4634	0.5620	0.0073	0.4693	0.5263	0.8172	0.2123	0.5321
0.2093	0.7254	0.2800	0.2093	0.0735	0.9038	0.0127	0.9083	0.2093	0.2282	0.4214	0.0612
0.2330	0.6656	0.3998	0.2330	0.0767	0.7412	0.0310	0.8206	0.2330	0.2333	0.4845	0.1411
0.4389	0.0168	0.9858	0.4389	0.9886	0.0504	0.0195	0.0782	0.4389	0.7806	0.0910	0.9200
0.0023	0.0025	0.0124	0.0023	0.0118	0.0012	0.0323	0.0010	0.0023	0.0003	0.0036	0.0021

En este caso, el modelo deja mucho que desear, porque si bien varios de los valores se acercan a su valor real, en muchos de los casos el valor es intermedio, lo cual dificulta interpretación y con esto la exactitud se vería bastante afectada.

Conclusión: Con esta práctica nos damos cuenta, que no siempre un esquema con mayor complejidad nos va a garantizar el mejor desempeño, pues teniendo en cuenta el primer caso, se observo que el modelo tuvo una exactitud perfecta en sus predicciones, sin embargo, al momento de realizar el entrenamiento del esquema 2 y 3 se usaron valores por defecto, en donde la tasa de aprendizaje u otros parámetros pueden afectar directamente. También otro factor a tener en cuenta es la poca cantidad de datos que se manejan para este propósito.