Universidad Simón Bolívar
Inteligencia Artificial II
Bernardo Morales, Leonardo Ramos y Rubén Serradas

1. Resumen

Se implementó una red neuronal multicapa feedforward y se probó su efectividad clasificando puntos en un plano pertenecientes a dos regiones predeterminadas: un rectángulo cuya diagonal es el segmento de recta que une los puntos (0,0) y (20,20) y otra región compuesta por un circulo de centro (10,10) y radio 7. Con los datos suministrados se entrenó la red y se probó esta con patrones realizados aleatoriamente de forma uniforme por toda el área para ver los resultados de este entrenamiento.

Por otro lado, se realizaron dos clasificadores mas con esta red sobre los datos del conjunto *Iris Data Set*: uno que separaba un tipo de planta Iris (Iris Setosa) de las demás del conjunto y otro que separase cada una de las 3 clases existentes.

2. Detalles de Implementación/Experimentación

Se utilizó el lenguaje C++ para la implementación de la red neuronal y los dos programas clasificadores `punto` y `iris`. Para la representación de la neurona se partió de la clase `Perceptron` y se extendió esta para que se comportará como una neurona. Es importante notar que en el metodo de la clase neurona `procesar_neurona` estamos usando la función tangente hiperbólica en vez de la función sigmoidal pues esta primera puede retornar valores negativos mientras que la segunda no. Luego de representar una neurona, representamos cada capa de la red mediante una clase llamada `Capa_red`. Por ultimo, tenemos la clase que representa a nuestra red `Red_neuronal`.

Para nuestro clasificador de puntos realizamos una red de 3 neuronas receptoras, 2 a 10 neuronas intermedias y una neurona de salida con una tasa de aprendizaje de 0.01, máxima iteración de 100, pesos de 1.0 a 5.0 y error mínimo de 0.01. Los resultados obtenidos se mostraran en la sección posterior.

Los clasificadores para el conjunto *Iris Data Set* están compuestos por 5 neuronas receptoras, 4 a 10 neuronas intermedias y 1 a 3 neuronas de salida (dependiendo del caso); se tiene una iteración máxima de 100000, tasa de aprendizaje de 0.05, error mínimo de 0.0001 y pesos de 1 a 4 para las neuronas. Para compilar ambos programas tan solo es necesario realizar `make`.

3. Resultados Obtenidos

3.1. Clasificador de puntos

Se realizaron pruebas con la red neuronal con datos proporcionados de 500, 1000 y 2000 instancias aleatorias y se probó la efectividad de la red con datos aleatorios realizados de 500, 1000 y 2000 puntos. Con los argumentos especificados anteriormente se obtuvo un mejor resultado con una red neuronal de 10 neuronas en la capa intermedia. Estos resultados se ven reflejados en la tabla 1.

Datos Prueba/Datos Entrenamiento	Error Total	Iteraciones Totales	Resultado (%)
500/500	0.009	91	96 %
500/1000	0.04	100	93 %
500/2000	0.09	100	96 %
1000/500	0.009	91	95 %
1000/1000	0.04	100	92 %
1000/2000	0.09	100	97 %
2000/500	0.009	91	95 %
2000/1000	0.04	100	91 %
2000/2000	0.09	100	97 %

Tabla 1: Tabla de resultados para una red neuronal de 10 neuronas en la capa intermedia.

Se observa como con mayor cantidad de datos, mayor es la precisión con que clasifica los puntos. Por otro lado, se puede observar los resultados del entrenamiento gráficamente mediante la imagen 1. Se realizó una prueba con 100*100 puntos en toda el área del rectángulo uniformemente separados y se coloreo un cuadrado con puntos negros cuando la red clasificaba un punto y con puntos blancos cuando clasificaba el rectángulo. Los errores obtenidos en la clasificación se observan en la imagen 2

Figura 1: Resultados de la prueba 100*100 con la red de 10 neuronas en la capa intermedia.

Figura 2: Errores de la prueba 100*100 con la red de 10 neuronas en la capa intermedia.

3.2. Clasificador de Iris

Para este experimento se tomaron como datos de 50 %, 60 %, 70 %, 80 % y 90 % del conjunto de datos *Iris Data Set*. Para que los conjuntos de entrena-

miento fueran adecuados se eligió del conjunto de 50 instancias de cada una de las Iris (setosa, versicolor y virginica) el porcentaje adecuado, de esta forma el clasificador tenia un conjunto de ejemplos mas equilibrado según lo que tenia que hacer. Los resultados del primer clasificador se ven reflejados en la tabla 2. Los resultados del clasificador de Iris se ven en **??**

Datos Prueba/Datos Entrenamiento (%)	Error Total	Iteraciones Totales	Resultado (%)
100/50	9.70154e-05	10	100 %
100/60	9.97123e-05	2	66 %
100/70	9.96403e-05	11	100 %
100/80	9.18244e-05	15	100 %
100/90	6.79173e-05	4	100 %

Tabla 2: Tabla de resultados para clasificador de Iris para una red neuronal de 10 neuronas en la capa intermedia.

Este primer clasificador con un conjunto de 50 % de los datos tiene una efectividad de 100 %. Es probable que con un mayor conjunto de datos de prueba se pueda comprobar mejor su aprendizaje, ya que con 90 % es casi el conjunto de prueba completo. El segundo valor obtenido en la tabla conjunto 60 % de conjunto de entrenamiento puede deberse a que es necesario un error mínimo mas pequeño ya que los datos aleatorios son grandes.

Datos Prueba/Datos Entrenamiento (%)	Error Total	Iteraciones Totales	Resultado (%)
100/50	9.99977e-05	80450	95 %
100/60	0.000316397	100000	98 %
100/70	0.000514852	100000	98,6 %
100/80	0.00489225	100000	99 %
100/90	9.02974e-05	20915	98 %

Tabla 3: Tabla de resultados para clasificador de Iris para una red neuronal de 10 neuronas en la capa intermedia.

Este segundo clasificador usa una mayor cantidad de iteraciones ya que se tiene 3 neuronas de salida. Su precisión es bastante grande, sin embargo el tiempo en que tarda este comparado con el primero es exorbitante. Una forma de mejorar esto podría ser realizando clasificadores binarios de cada uno de las Iris y combinarlos para obtener una mejora del tiempo y una buena precisión.