PRÁCTICA 2 DE NEUROCOMPUTACIÓN
Curso 2013-2014

Introducción a las Redes Neuronales Artificiales

Publicación: 3 de Marzo. Entrega: 28 de Marzo

1. Introducción

Las redes neuronales estudiadas hasta ahora, es decir, el Perceptrón y la red ADALINE, contienen una sola capa y están por lo tanto limitadas, en el sentido de que sólo pueden resolver problemas linealmente separables. Una forma sencilla de abordar problemas que no son separables linealmente consiste en introducir una capa extra de neuronas no lineales, llamada capa oculta, entre la capa de entrada y la capa de salida. El problema que surge a continuación consiste en cómo entrenar dicha red multicapa. Una solución es el algoritmo de retropropagación de errores. Dicho algoritmo consta de dos fases bien diferenciadas:

1. Propagación hacia delante: Se introduce un patrón en la entrada de la red como estímulo, y se propagan las activaciones de las distintas neuronas desde la primera capa hasta capa de salida para generar la salida de la red.

2. Retro-propagación de los errores: La señal de salida generada se compara con la salida deseada y se calcula un valor de error. Dicho error se propaga hacia atrás, partiendo de la capa de salida, hacia la capa oculta de neuronas. Basándose en la señal de error recibida, se actualizan los pesos de conexión de cada neurona, para minimizar el error cuadrático entre la salida generada por la red y la salida deseada.

Este proceso, iterado múltiples veces para cada patrón de entrenamiento, fuerza a que las neuronas de la red aprendan a reconocer distintas características de los datos de entrenamiento que se puedan utilizar para la clasificación correcta de nuevos patrones no observados por la red. Es decir, la red es capad de extraer cierto conocimiento de los datos de entrenamiento y utilizarlo para generalizar.

2. Objetivos

Los objetivos de esta tercera práctica son la implementación y análisis de una red neuronal multi-capa entrenada mediante el algoritmo de retro-propagacón de errores. Las tareas a realizar son las siguientes:

Tarea 1: implementación de la red neuronal.
Implementar de forma genérica el algoritmo de retro-propagación de errores para entrenar una red con una sola capa oculta y funciones de activación sigmoide bipolar (en capas oculta y de salida):

$$f(x) = \frac{2}{1+\exp(-x)} - 1\,, \qquad f'(x) = \frac{1}{2}\left(1+f(x)\right)\left(1-f(x)\right)\,.$$

El número de neuronas de salida será igual al de número de clases.

Cuando se use la red para clasificación, la clase predicha por la red corresponde a la neurona con mayor activación.

El programa recibirá como argumentos:

- El nombre del fichero con los datos del problema
- El % del conjunto de training
- La tasa de aprendizaje
- El número de neuronas de la capa oculta.
- El nombre de un fichero en el que el programa escribirá en cada línea el error cuadrático medio de la red para cada época, así como los valores de los pesos de la red. Entenderemos que ha transcurrido una época cuando todos los patrones de entrenamiento hayan sido utilizados para actualizar los pesos de la red.
- Finalmente, el programa imprimirá por pantalla las tasas de acierto obtenidas en el conjunto de entrenamiento y test. El entrenamiento terminará una vez hayan transcurrido 1, 000 épocas.

Tarea 2: chequeo del funcionamiento de la red con los ficheros de la práctica anterior.
Entrenar una red neuronal multi-capa con las mismas bases de datos de la anterior práctica. Se deberán probar diferente número de neuronas en la capa oculta.

Se deberán hacer gráficas de los errores en training/test en función de la época, para diferente numero de neuronas en la capa oculta, y diferentes valores de la constante de aprendizaje.

Tarea 3: predicción en problemas con más de dos clases.
Usa lo desarrollado en la parte anterior para obtener un modelo de predicción para la base de datos **problema_real_3clases.txt**
Haz el mismo tipo de gráficas y análisis que en la parte anterior

Tarea 4: predicción en un problema complejo.
Haz lo mismo con la base de datos **problema_real4.txt**
¿Qué tasa de error obtienes?
Haz una estadística de los atributos de entrada del problema (medias y desviaciones estándar de cada uno de los 9 atributos). ¿Tienen una estadística apropiada para la red neuronal que estás usando?

Tarea 5: normalización de los datos.
Repetir los experimentos del anterior apartado, pero esta vez normalizando previamente los datos. Es decir, restando a cada atributo su media a lo largo de **todos los patrones de entrenamiento** y dividiendo por su desviación estándar (**calculada también con sólo los patrones del conjunto de entrenamiento**). Por ejemplo, sea x_{ji} el atributo i-ésimo correspondiente al patrón j-ésimo de entrenamiento. Tras el proceso de normalización, reemplazaremos x_{ji} por:

$$x_{ji}^{nuevo} = \frac{x_{ji} - \hat{\mu}_i}{\hat{\sigma}_i}$$

Donde $\hat{\mu}_i$ es la media del atributo i-ésimo estimada con los datos de entrenamiento, y $\hat{\sigma}_i$ es la desviación estándar del atributo i-ésimo, estimada con los datos de entrenamiento.

Compara los resultados obtenidos con los de la Tarea 4.

Tarea 6: predicción de datos no etiquetados.
Finalmente, se desarrollará un modelo para el problema de clasificación definido en la base de datos **problema_real2.txt**. Se deberá generar un fichero **predicciones_nnet.txt** con las predicciones realizadas por dicho modelo para los datos no etiquetados definidos en:

problema_real2_no_etiquetados.txt

El formato del fichero **predicciones_nnet.txt** será texto plano, donde habrá tantas filas como número de patrones en

problema_real2_no_etiquetados.txt, y donde en cada línea habrá solamente 1 0 o 0 1 dependiendo de la clase predicha.

3. Material a entregar

Se entregará una memoria en formato pdf con los análisis de las cuestiones planteadas en el enunciado y también contendrá los resultados de los distintos experimentos descritos y las comparaciones pedidas.

Se entregará el código fuente junto con el Makefile, fichero help, etc.

Se entregará el fichero generado **predicciones_nnet.txt**

Referencias

Neural networks. A comprehensive foundation. Simon Haykin. Prentice Hall. 1999. Neural Networks for Pattern Recognition. C.M. Bishop. Oxford University Press 1996.

Redes Neuronales Artificiales. Un enfoque pr´actico. P. Isasi, I.M. Galv´an. Pearson Prentice Hall 2004.