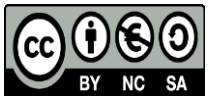
uc3m | Universidad Carlos III de Madrid

OPENCOURSEWARE
REDES DE NEURONAS ARTIFICIALES
Inés M. Galván – José M. Valls



Tema 3: Perceptron Multicapa

En este tema se presenta la red de neuronas conocida como Perceptron Multicapa (PM). Dados unos aspectos introductorios, se explica en primer lugar los detalles relativos a como es la arquitectura, es decir cómo se organizan las neuronas y conexiones en la red, cómo se calculan las activaciones de las neuronas de la red (regla de propagación) y la función de activación utilizada.
A continuación se presenta el objetivo del aprendizaje de la red, concretamente minimizar el error medido en la salida de la red, para deducir la regla de aprendizaje, que para el PM recibe el nombre de regla delta generalizada o algoritmo de retropropagación. Las transparencias de desde la 19 hasta la 31 se dedican a la deducción del algoritmo de aprendizaje, mostrando un resumen de la regla en las transparencias 32. Para deducir la regla de aprendizaje es necesario distinguir si el peso a modificar es un peso de la última capa (transparencias 19-21) o de las capas anteriores (transparencias 22-29). Si el lector prefiere no estudiar la deducción de la regla delta generalizada puede obviar las transparencias de 19 a 31 y estudiar la transparencia 32 donde se muestra un resumen de las ecuaciones de la regla de aprendizaje.
Una vez estudiada la regla de aprendizaje, se muestran los pasos para realizar el aprendizaje o entrenamiento de una arquitectura de PM. Se incluyen también algunos comentarios sobre el efecto de la razón e aprendizaje, parámetro que interviene en dicho proceso.
El resto del tema se dedica a ciertas cuestiones relacionadas con esta red y su uso práctico. En primer lugar, se analiza cómo se puede evitar en este tipo de red el sobreajuste. Se explican también algunos de los problemas más habituales que se pueden encontrar durante el aprendizaje (mínimos locales y parálisis), y se presentan posibles estrategias para poder evitar estos problemas.
Finalmente, se presentan de manera general los problemas de clasificación y regresión no lineal, indicando como abordarlos con un PM.
En lo que sigue se presenta un resumen del contenido de las transparencias y al final de este documento se incluye aspectos importantes a la hora de utilizar el PM para resolver un problema dado.

3.1 Introducción

El Perceptron Multicapa (PM) fue propuesta por Rumelhart y otros autores (1986) y surgió con el objetivo de solventar las limitaciones encontradas en el Perceptron Simple frente a su incapacidad para resolver problemas no lineales. Se trata de una red feed-forward, es decir con conexiones hacia adelante y el aprendizaje de la red es supervisado. Puede utilizarse para aproximar relaciones no lineales entre variables de entrada y salida y se conoce como un aproximador universal, en el sentido que puede aproximar cualquier función continua.

Hoy en día, el PM es una de las arquitecturas de redes de neuronas más utilizada y conocida, hasta el punto que en muchas ocasiones se utiliza el término red de neuronas para referirse al PM.

3.2 Arquitectura del PM

La arquitectura del PM (ver figura en transparencia 7) se caracteriza porque las neuronas están organizados en distintas capas: entrada, ocultas y salida. La capa de entrada está formada por las neuronas de entrada, que sólo se encargan de recibir las señales de entrada y propagarlas a la siguiente capa. Las capas ocultas realizan un procesamiento no lineal de los datos recibidos en la entrada y la capa de salida está formada por las neuronas de salida. Se caracteriza además porque todas las neuronas de una capa están conectadas a todas las neuronas de la siguiente capa. Y cada una de estas conexiones tiene un número real asociado, llamado peso de la conexión.

La activación de las neuronas de entradas viene dado por el patrón de entrada. La activación del resto de las neuronas de la red se calcula teniendo en cuenta las activaciones de las neuronas de la capa anterior multiplicada por los pesos correspondientes (ver transparencias 9 y 10).

En el PM la función de activación para las neuronas de las capas ocultas es la función sigmoidal, y es justo el uso de esta función en la activación de las neuronas ocultas lo que marca el carácter no lineal del PM. Las neuronas de salida pueden tener función de activación sigmoigal, pero también la función lineal f(x)=x. Este función lineal nunca puede utilizarse para las neuronas ocultas, pues estaríamos entonces ante un Percpetron Simple.

Debido al uso de la función de activación no lineal, el PM define una relación no lineal entre las variables de entrada y salida. Esta relación se obtiene propagando hacia la capa de salida, pasando por las capas ocultas, los valores de las variables de entrada y puede representarse como:

$$Y=F(X, W) \qquad (1)$$

siendo X el vector de variables de entrada y W los pesos y umbrales de la red.

3.3 Aprendizaje del PM: Algoritmo de retropropagación o Regla Delta generalizada

El aprendizaje del PM (procedimiento para determinar los pesos y umbrales de la red) es supervisado, por lo que para cada patrón de entrada es necesario disponer de un vector o valor de salida deseada. De este modo, el aprendizaje consiste en **minimizar el error** que se mide en la salida de la red (error cuadrático medio) (ver transparencia 16). Por tanto, los pesos y umbrales del PM se van ajustando de manera paulatina utilizando el método del descenso del gradiente, es decir la derivada negativa del error:

$$w(n) = w(n-1) - \alpha \frac{\partial e(n)}{\partial w} \qquad (2)$$

Para poder entonces ajustar los pesos es necesario calcular la deriva del error respecto a cada peso, $\frac{\partial e(n)}{\partial w}$. Al hacer este cálculo, se obtiene la regla delta generalizada o algoritmo de retropropagación, que básicamente consiste en actualizar cada peso de la red multiplicando **la activación de la neurona de la parte el peso y el delta de la neurona a la que llega el peso**. Así, dado el peso w_{kj}^{c} que une la neurona k de la capa c con la neurona j de la capa c+1, hay que tener en cuenta la activación de la neurona k de la capa c y el delta neurona j de la capa c+1, es decir:

$$w_{kj}^{c}(n) = w_{kj}^{c}(n-1) + \alpha \delta_{j}^{c+1}(n) a_{k}^{c}(n) \quad (3)$$

La expresión del **término delta** difiere si se trata de un peso de la última capa o de las capas anteriores. Para los pesos de la última capa, el delta mide el error que se comente en la neurona de salida y tiene en cuanta la derivada de la función de activación para las neuronas de salida:

$$\delta_i^C = (s_i - y_i) y_i (1 - y_i) \quad (4)$$

Nótese que si se utiliza la función de activación lineal en la salida, la expresión del término delta viene dado por:

El término delta para el resto de las neuronas de la red se calcula utilizando la derivada de su función de activación y la suma de los términos δ de las neuronas de la siguiente capa ponderados por los pesos correspondientes, es decir:

$$\delta_j^{c+1} = a_j^{c+1}(1 - a_j^{c+1}) \sum_{i=1}^{n_{c+2}} \delta_i^{c+2} w_{ji}^{c+1} \quad (5)$$

De ahí que se llame algoritmo de retropropagación, pues los errores medidos en la salida de la red se retropropagan hacia atrás.

Dada una arquitectura de PM, los pasos para realizar el **entrenamiento o aprendizaje** se presentan las transparencias 36 y 37. La idea intuitiva de este proceso es: Partiendo de un conjunto de pesos aleatorios, W(0), en la superficie del error, la regla de aprendizaje desplaza dicho vector de pesos siguiendo la dirección negativa del gradiente del error en dicho punto, alcanzando un nuevo punto W(1) que estará más próximo del mínimo del error que el anterior. Partiendo ahora de W(1), el vector de pesos se desplaza obteniendo W(2) y así sucesivamente. El aprendizaje finaliza cuando se encuentra un mínimo, ya que en este punto la derivada es cero y según la ecuación (2) los pesos dejan de modificarse.

La razón de aprendizaje (valor entre 0 y 1), al igual que en el resto de las redes de neuronas, mide la magnitud del desplazamiento. Razones de aprendizaje altas favorecen a una convergencia más rápida, pero con el peligro de oscilar alrededor del mínimo. Razones más pequeñas hacen que la convergencia sea más lenta. El valor más adecuado de la razón de aprendizaje depende el problema, pues depende de la forma de la superficie del error. Dicho parámetro se determina experimentalmente.

3.4 Algunos problemas habituales en el uso del PM

Escasa capacidad de generalizan: En el PM, en ocasiones una baja capacidad de generalización viene dada por el uso de un gran número de neuronas ocultas. Si existen pocos patrones y el número de neuronas ocultas es elevado, podría ocurrir que la red se especialice en los patrones de entrenamiento.

Mínimo local: Debido a que los pesos de la red se modifican teniendo en cuanta la derivada del error, el aprendizaje puede caer en un mínimo local, y no proporcionar una solución aceptable al problema. Algunas posibles soluciones para intentar conseguir un mejor mínimo son: aumentar el número de neuronas ocultas o partir de diferentes inicializaciones aleatorias de los pesos, ente otras.

Parálisis: La parálisis en el PM viene dado porque los pesos toman valores muy altos (positivos o negativos) lo cual hace que la activación de las neuronas tome los valores de las asíntotas de la sigmoide y la red deje de aprender. Para evitar este fenómeno es conveniente normalizar en el intervalo [0,1] los patrones de entrada y salida deseada de la red, así como inicializar los pesos y umbrales de la red con valores cercanos a cero.

3.5 Diseño de un PM para abordar un problema

El diseño de un PM para un problema dado implica determinar:

- **Número de neuronas de entrada y salida**: Vienen dados por el problema a resolver. Para un problema de regresión, las neuronas de entrada y salida son las variables de entrada y salida. Para un problema de clasificación, el número de neuronas de entrada viene dada por los atributos de entrada del problema, y el número de neuronas de salida vienen dada por la codificación empleada (ver transparencia 48)

- **Número de capas ocultas y de neuronas en cada capa**: Existe un resultado teórico que demuestra que con una única capa oculta se puede encontrar una posible solución al problema. Sin embrago, este resultado no dice cuántas neuronas ocultas son necesarias, por lo que en ocasiones dos o más capas ocultas pueden proporcionar una mejor solución.

El número de capas y de neuronas ocultas se determinan experimentalmente, pero el diseñador debe tener en cuenta que dado un problema, existe una gran variedad de posibles arquitecturas que proporcionen una solución similar, por lo que el proceso de búsqueda se reduce bastante. Se aconseja, partir de un PM con única capa oculta, y realizar un conjunto de experimentos variando el número de neuronas en dicha capa, pero probando con números distantes (por ejemplo 10, 20, 30, …, pero nunca 10, 11, 12, …). Si se desea, una vez localizado el mejor número de neuronas ocultas, se puede realizar algún experimento utilizando dos capas ocultas y con una cantidad equivalente de neuronas ocultas.

- **Parámetros del aprendizaje: razón de aprendizaje y número de ciclos**. Ambos parámetros están relacionados porque mayor razón de aprendizaje, menor número de ciclos y al contrario. Por tanto, para cada razón de aprendizaje es necesario determinar el número de ciclos más adecuado. Es necesario también tener presente que cada arquitectura de PM puede requerir una razón de aprendizaje y/o número de ciclos diferente. Por tanto, fijado una arquitectura (número de neuronas ocultas) se aconseja realizar una batería de experimentos para determinar ambos parámetros, del siguiente modo:

RA1: ciclos=100, ciclos=200, ciclos=500, …,ciclos=Max
RA2: ciclos=100, ciclos=200, ciclos=500, …,ciclos=Max
RA3: ciclos=100, ciclos=200, ciclos=500, …,ciclos=Max

Para elegir la mejor configuración de parámetros, se aconseja utilizar un conjunto de validación, independiente del conjunto de entrenamiento y test.

Es también importante tener presente que los valores más adecuados para el número de neuronas ocultas, razón de aprendizaje y número de ciclos dependen del problema. Para un problema, 10 neuronas ocultas pueden ser muchas, pero para otro, pocas. Igualmente para un problema, 500 ciclos pueden ser suficientes, pero para otros requerir miles de ciclos de aprendizaje.