Introducción a los modelos computacionales

Práctica 1. Implementación del perceptrón multicapa

Javier Sánchez Monedero (jsanchezm@uco.es)
Pedro Antonio Gutiérrez (pagutierrez@uco.es)

Asignatura "Introducción a los modelos computacionales"
4º Curso Grado en Ingeniería Informática
Especialidad Computación
Escuela Politécnica Superior
(Universidad de Córdoba)

1 de octubre de 2024

Objetivos de la práctica

- Familiarizar al alumno con los modelos computacionales de redes neuronales, en concreto, con el perceptrón multicapa.
- Implementar el algoritmo de retropropagación básico para el perceptrón multicapa.
- Comprobar el efecto de distintos parámetros y preprocesamiento:
  - Arquitectura de la red.
  - Factor de momento.
  - Normalización de datos.
  - etc.

Algoritmo de retropropagación

- Leer y analizar los apuntes de teoría.
- Analizar especialmente el pseudocódigo.

Condición de parada

- Versión estándar, el algoritmo para si:
  - El error de entrenamiento no baja más de 0,00001 o sube, durante 50 iteraciones (bucle externo).

Consideraciones para la normalización de datos I

- La mayoría de métodos de aprendizaje automático necesitan que los datos estén normalizados para mitigar el efecto de las diferentes magnitudes de las variables.
- Existen muchas variantes de la normalización, pero las dos más comunes son:
  - Escalado: cada variable se transforma para que esté en un intervalo $[a, b]$:

$$X' = a + \frac{(X - X_{\text{mín}})\,(b - a)}{X_{\text{máx}} - X_{\text{mín}}} \tag{1}$$

  - Estandarización: normaliza cada variable típicamente produciendo una distribución de media cero y desviación típica 1:

$$X' = \frac{X - \mu}{\sigma} \tag{2}$$

Consideraciones para la normalización de datos II

- En ambos casos es importante que los parámetros de escalado se calculen sobre el conjunto de entrenamiento, y luego se apliquen las transformaciones en el conjunto de entrenamiento y en el test.
- Un error común es aplicar la normalización sobre el conjunto de test calculando los valores mínimos y máximos ($X_{\text{máx}}$ y $X_{\text{mín}}$), o la media y desviación (μ y σ), sobre este conjunto o antes de realizar la partición en conjuntos de entrenamiento y test. En ambos casos se está utilizando directa o indirectamente información de test para la construcción del modelo.
- En nuestro caso, vamos a escalar las variables de entrada en el rango $[-1, 1]$ y las variables de salida en $[0, 1]$ ya que la última capa de salida es una sigmoide.

Algoritmo de retropropagación

Algoritmo de retropropagación *on-line*

Inicio

1. $w_{ji}^h \leftarrow U[-1,1]$ *// Aleatorios entre -1 y $+1$*
2. **Repetir**
   1. **Para** cada patrón con entradas **x**, y salidas **d**
      1. $\Delta w_{ji}^h \leftarrow 0$ *// Se aplicarán cambios por cada patrón*
      2. $out_j^0 \leftarrow x_j$ *// Alimentar entradas*
      3. forwardPropagation() *// Propagar las entradas ($\Rightarrow\Rightarrow$)*
      4. backPropagation() *// Retropropagar el error ($\Leftarrow\Leftarrow$)*
      5. accumulateChange() *// Calcular ajuste de pesos*
      6. weightAdjustment() *// Aplicar el ajuste calculado*

      Fin Para

   Hasta (CondicionParada)
3. **Devolver** matrices de pesos.

Fin

Algoritmo de retropropagación

Algoritmo de retropropagación *off-line*

Inicio

1. $w_{ji}^{h} \leftarrow U[-1,1]$ // *Aleatorios entre* -1 *y* $+1$
2. **Repetir**
   1. $\Delta w_{ji}^{h} \leftarrow 0$ // *Se aplicarán cambios al final*
   2. **Para** cada patrón con entradas $\mathbf{x}$, y salidas $\mathbf{d}$
      1. $out_j^0 \leftarrow x_j$ // *Alimentar entradas*
      2. forwardPropagation() // *Propagar las entradas* ($\Rightarrow\Rightarrow$)
      3. backPropagation() // *Retropropagar el error* ($\Leftarrow\Leftarrow$)
      4. accumulateChange() // *Acumular ajuste de pesos*

      Fin Para
   3. weightAdjustment() // *Aplicar el ajuste calculado*

   Hasta (CondicionParada)
3. **Devolver** matrices de pesos.

Fin

Algoritmo de retropropagación: funciones

forwardPropagation()

Inicio

1. **Para** h de 1 a H *// Para cada capa* $(\Rightarrow\Rightarrow)$
   1. **Para** j de 1 a n_h *// Para cada neurona de la capa h*
      1. $net_j^h \leftarrow w_{j0}^h + \sum_{i=1}^{n_{h-1}} w_{ji}^h out_i^{h-1}$
      2. $out_j^h \leftarrow \frac{1}{1+\exp(-net_j^h)}$

      Fin Para

   Fin Para

Fin

Algoritmo de retropropagación: funciones

backPropagation()

Inicio

1. **Para** j de 1 a n_H *// Para cada neurona de salida*
    1. $\delta_j^H \leftarrow -\left(d_j - out_j^H\right) \cdot g'(net_j^H)$ *// Hemos obviado la constante (2), ya que el resultado será el mismo*

    Fin Para
2. **Para** h de $H-1$ a 1 *// Para cada capa ($\Leftarrow\Leftarrow$)*
    1. **Para** j de 1 a n_h *// Para cada neurona de la capa h*
        1. $\delta_j^h \leftarrow \left(\sum_{i=1}^{n_{h+1}} w_{ij}^{h+1}\delta_i^{h+1}\right) \cdot out_j^h \cdot (1 - out_j^h)$ *// Pasa por todas las neuronas de la capa $h+1$ conectadas con j*

        Fin Para

    Fin Para

Fin

Algoritmo de retropropagación: funciones

acummulateChange()

Inicio

1. **Para** h de 1 a H *// Para cada capa (⇒⇒)*
   1. **Para** j de 1 a n_h *// Para cada neurona de la capa h*
      1. **Para** i de 1 a n_{h-1} *// Para cada neurona de la capa $h-1$*
         $\Delta w_{ji}^h \leftarrow \Delta w_{ji}^h + \delta_j^h \cdot out_i^{h-1}$
         Fin Para
      2. $\Delta w_{j0}^h \leftarrow \Delta w_{j0}^h + \delta_j^h \cdot 1$ *// Sesgo*

      Fin Para

   Fin Para

Fin

Algoritmo de retropropagación: funciones

weightAdjustment()

Inicio

1. **Para** h de 1 a H *// Para cada capa* ($\Rightarrow\Rightarrow$)
    1. **Para** j de 1 a n_h *// Para cada neurona de la capa h*
        1. **Para** i de 1 a n_{h-1} *// Para cada neurona de la capa* $h-1$
           $w_{ji}^h \leftarrow w_{ji}^h - \eta\Delta w_{ji}^h - \mu\left(\eta\Delta w_{ji}^h(t-1)\right)$
           Fin Para
        2. $w_{j0}^h \leftarrow w_{j0}^h - \eta\Delta w_{j0}^h - \mu\left(\eta\Delta w_{j0}^h(t-1)\right)$ *// Sesgo*

       Fin Para

   Fin Para

Fin

Introducción a los modelos computacionales

Práctica 1. Implementación del perceptrón multicapa

Javier Sánchez Monedero (jsanchezm@uco.es)
Pedro Antonio Gutiérrez (pagutierrez@uco.es)

Asignatura "Introducción a los modelos computacionales"
4º Curso Grado en Ingeniería Informática
Especialidad Computación
Escuela Politécnica Superior
(Universidad de Córdoba)

1 de octubre de 2024