Extreme Learning Machine

Modelos de la computación (Aprendizaje supervisado)

Francisco Fernández Navarro

Departamento de Lenguajes y Ciencias de la Computación
Área: Ciencias de la Computación e Inteligencia Artificial

Índice de contenidos

¿Qué es Extreme Learning Machine (ELM)?

Definición

Extreme Learning Machine (ELM) es un algoritmo de aprendizaje automático que se utiliza para entrenar redes neuronales artificiales de una sola capa oculta de manera eficiente y rápida.

- Fue propuesto por Huang, Wang y Lan en 2006 como una alternativa eficiente a los métodos de entrenamiento tradicionales de redes neuronales.
- ELM se destaca por su velocidad de entrenamiento y su capacidad para manejar grandes conjuntos de datos.
- Se utiliza en diversas aplicaciones, incluyendo clasificación, regresión y reconocimiento de patrones.

Características Clave de ELM

1. **Capa Oculta Aleatoria:** En ELM, la capa oculta se inicializa de forma aleatoria con valores de peso y sesgo, lo que acelera el entrenamiento.
2. **Entrenamiento Rápido:** ELM no requiere ajuste iterativo de pesos, lo que lo hace mucho más rápido que otros métodos de entrenamiento.
3. **Generalización Sólida:** A pesar de su entrenamiento rápido, ELM puede generalizar bien a partir de datos de entrenamiento ruidosos.
4. **Aplicaciones Versátiles:** Se utiliza en una amplia gama de aplicaciones de aprendizaje automático, desde reconocimiento de voz hasta procesamiento de imágenes.

Diferencias entre ELM y Redes Backpropagation

Entrenamiento

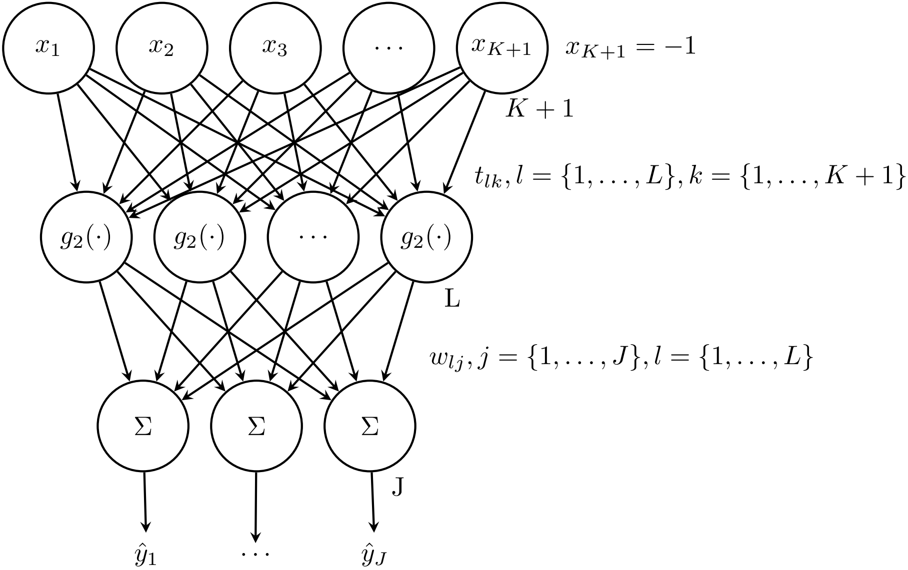
- ELM utiliza un entrenamiento de tipo **feedforward**.
- No hay ajuste iterativo de pesos.
- Inicialización aleatoria de la capa oculta.
- Cálculo directo de los pesos de la capa de salida.

Arquitectura

- ELM tiene **una sola capa oculta**.
- ELM tiende a utilizar **muchas neuronas** en la capa oculta.
- La cantidad de neuronas en la capa oculta **no es crítica** para el rendimiento.
- **No hay sesgo en la capa de salida**.
- En redes tradicionales, múltiples capas ocultas y sesgo en cada capa.

Índice de contenidos

Misma nomenclatura a la usada en el tema anterior (redes MLP).



Nomenclatura Extreme Learning Machine

Clasificación multi-clase

Los parámetros del ELM se estiman a partir de un conjunto de entrenamiento $\mathcal{D} = (\mathbf{X}, \mathbf{Y}) = \{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^{N}$, donde $\mathbf{x}_n = (x_{n1}, x_{n2}, \ldots, x_{nK}) \in \mathbb{R}^K$ es el vector de atributos del n-ésimo patrón, K es la dimensión del espacio de entrada (número de atributos en el problema), $\mathbf{y}_n \in \{0,1\}^J$ es la etiqueta de clase asumiendo la codificación "1-de-J" ($y_{nj} = 1$ si $\boldsymbol{x}_n$ es un patrón de la j-ésima clase, $y_{nj} = 0$ en caso contrario), y J es el número de clases. Denotemos $\mathbf{Y}$ como $\mathbf{Y} = (\mathbf{Y}_1 \ldots \mathbf{Y}_J) = \begin{pmatrix} \mathbf{y}_1' \\ \vdots \\ \mathbf{y}_N' \end{pmatrix} \in \{0,1\}^{N \times J}$, donde $\mathbf{Y}_j$ es la j-ésima columna de la matriz $\mathbf{Y}$. La función $f : \mathbb{R}^K \rightarrow \{0,1\}^J$.

Nomenclatura Extreme Learning Machine

Parámetros

En ELM tenemos que estimar los valores de dos matrices:

- $\mathbf{w} = (\mathbf{w}_1 \ \mathbf{w}_2 \ \ldots \mathbf{w}_J) \in \mathbb{R}^{L \times J}$: Matriz con los valores de los pesos sinápticos de las neuronas que conectan capa de salida con capa oculta. $\mathbf{w}_j \in \mathbb{R}^L$ son las conexiones de la neurona de salida j con la capa oculta, $\forall j \in \{1, \ldots, J\}$.
- $\mathbf{t} = (\mathbf{t}'_1, \mathbf{t}'_2, \ldots, \mathbf{t}'_L) \in \mathbb{R}^{L \times (K+1)}$: Matriz con los valores de los pesos sinápticos de las neuronas que conectan capa oculta con capa de entrada. $\mathbf{t}_l \in \mathbb{R}^{K+1}$ son las conexiones de la neurona oculta l con la capa de entrada, $\forall l \in \{1, \ldots, L\}$.

Importante

Observar por el tamaño de las dos matrices que el sesgo únicamente ha sido incluido en la capa oculta.

Modelo funcional

La salida de ELM para un patrón n-ésimo, $\mathbf{x}_n \in \mathbb{R}^K$ es:

$$\hat{y}_{nj} = \sum_{l=1}^{L} w_{lj} s_{ln} = \sum_{l=1}^{L} w_{lj} \left(\mathrm{g}_2 \left(\sum_{k=1}^{K+1} t_{lk} x_{nk} \right) \right). \tag{1}$$

Definición de variables

- $\hat{y}_{nj}$: Estimación de salida de la clase j para el patrón n.
- s_{ln}: Salida de la neurona en capa oculta l para el patrón n.
- $\mathrm{g}_2(\mathbf{t}_l, \mathbf{x}_n)$: Función de transferencia de capa oculta.

Importante

Observar que no hay función de transferencia en la capa de salida.

Modelo funciónal - Formato matricial

Modelo funcional matricial

En formato matricial, la salida de ELM para un patrón n-ésimo, $\mathbf{x}_n \in \mathbb{R}^K$ es

$$\hat{\mathbf{y}}(\mathbf{x}_n) = \mathbf{h}'(\mathbf{x}_n)\,\mathbf{w},$$

Definición de variables

- $\mathbf{h} : \mathbb{R}^K \to \mathbb{R}^L$ es la función de mapeo de la capa de entrada a la capa oculta. $\mathbf{h}(\mathbf{x}_n) = (\mathrm{g}_2(\mathbf{t}_1, \mathbf{x}_n), \ldots, \mathrm{g}_2(\mathbf{t}_L, \mathbf{x}_n)) \in \mathbb{R}^L$.

Matriz de salidas en capa oculta

También denotamos $\mathbf{H}$ como $\mathbf{H} = (\mathbf{h}'(\mathbf{x}_n), \forall i \in \{1, \ldots, n\}) \in \mathbb{R}^{N\times L}$, la transformación del conjunto de entrenamiento desde el espacio de entrada al espacio transformado. Alternativamente puede expresarse como:

$$\mathbf{H} = \begin{pmatrix} \mathrm{g}_2(\mathbf{t}_1, \mathbf{x}_1) & \mathrm{g}_2(\mathbf{t}_2, \mathbf{x}_1) & \ldots & \mathrm{g}_2(\mathbf{t}_L, \mathbf{x}_1) \\ \mathrm{g}_2(\mathbf{t}_1, \mathbf{x}_2) & \mathrm{g}_2(\mathbf{t}_2, \mathbf{x}_2) & \ldots & \mathrm{g}_2(\mathbf{t}_L, \mathbf{x}_2) \\ \ldots & \ldots & \ldots & \ldots \\ \mathrm{g}_2(\mathbf{t}_1, \mathbf{x}_N) & \mathrm{g}_2(\mathbf{t}_2, \mathbf{x}_N) & \ldots & \mathrm{g}_2(\mathbf{t}_L, \mathbf{x}_N) \end{pmatrix} \in \mathbb{R}^{N\times L}$$

Índice de contenidos

Extreme Learning Machine

ELM minimiza el siguiente problema de optimización:

$$\min_{\mathbf{w}\in\mathbb{R}^{L\times J}} \left(\|\mathbf{w}\|^2 + C\|\mathbf{Hw} - \mathbf{Y}\|^2 \right), \tag{2}$$

donde $C \in \mathbb{R}$ es un parámetro especificado por el usuario que promueve el rendimiento de generalización.

Importante

- Problema conocido como Ridge Regression.
- La minimización de la norma de los parámetros actúa como una técnica de regularización que ayuda a prevenir el sobreajuste.
- Al penalizar los parámetros grandes (mediante la norma), se fomenta la simplicidad del modelo, lo que a menudo mejora su capacidad de generalización.
- La regularización de ELM es de tipo l_2.

Extreme Learning Machine

El problema de optimización puede reformularse como:

$$\begin{aligned} \|\mathbf{w}\|^2 + C\|\mathbf{H}\mathbf{w} - \mathbf{Y}\|^2 &= \mathbf{w}'\mathbf{w} + C(\mathbf{w}'\mathbf{H}' - \mathbf{Y}')(\mathbf{H}\mathbf{w} - \mathbf{Y}) \\ &= \mathbf{w}'\mathbf{w} + C\mathbf{w}'\mathbf{H}'\mathbf{H}\mathbf{w} - 2C\mathbf{Y}'\mathbf{H}\mathbf{w} + \mathbf{Y}'\mathbf{Y} \end{aligned} \tag{3}$$

Por lo tanto, el problema de optimización se puede reescribir como:

$$\min_{\mathbf{w}\in\mathbb{R}^{L\times J}} \left(\mathbf{w}'(\mathbf{I} + C\mathbf{H}'\mathbf{H})\mathbf{w} - 2C\mathbf{Y}'\mathbf{H}\mathbf{w}\right), \tag{4}$$

ya que $\mathbf{Y}'\mathbf{Y}$ es constante con respecto a $\mathbf{w}$.

Índice de contenidos

Extreme Learning Machine

La solución final se obtiene derivando el problema de optimización e igualando a cero y queda de la siguiente manera:

$$\mathbf{w} = \left(\frac{\mathbf{I}}{C} + \mathbf{H}'\mathbf{H} \right)^{-1} \mathbf{H}'\mathbf{Y} \tag{5}$$

donde **I** es la matriz identidad. La solución puede escribirse alternativamente como:

$$\mathbf{w} = \mathbf{H}' \left(\frac{\mathbf{I}}{C} + \mathbf{H}\mathbf{H}' \right)^{-1} \mathbf{Y} \tag{6}$$

Importante

Esta segunda versión es la que se utilizará en la versión kernel.

Extreme learning machine

Lemma

$$\mathbf{H}'\left(\frac{\mathbf{I}}{C}+\mathbf{H}\mathbf{H}'\right)^{-1}\mathbf{Y}=\left(\frac{\mathbf{I}}{C}+\mathbf{H}'\mathbf{H}\right)^{-1}\mathbf{H}'\mathbf{Y} \tag{7}$$

Proof.

Debemos probar que las matrices a la izquierda de $\boldsymbol{Y}$ coinciden:

$$\mathbf{H}'\left(\frac{\mathbf{I}}{C}+\mathbf{H}\mathbf{H}'\right)^{-1}=\left(\frac{\mathbf{I}}{C}+\mathbf{H}'\mathbf{H}\right)^{-1}\mathbf{H}' \tag{8}$$

Multiplicando ambos lados por las matrices inversas correspondientes, la proposición quedará demostrada si:

$$\left(\frac{\mathbf{I}}{C}+\mathbf{H}'\mathbf{H}\right)\mathbf{H}'=\mathbf{H}'\left(\frac{\mathbf{I}}{C}+\mathbf{H}\mathbf{H}'\right) \tag{9}$$

lo cual es trivial. □

Índice de contenidos

Extreme Learning Machine

Existen dos posibles implementaciones del marco de trabajo ELM: la red neuronal y la versión del kernel. La principal diferencia entre estos dos enfoques radica en la forma de calcular la función $\mathbf{h}(\mathbf{x})$ (para el patrón $\mathbf{x}$) y, en consecuencia, la matriz $\mathbf{H}$. En la implementación neuronal del marco de trabajo, $\mathbf{h}(\mathbf{x})$ se puede calcular explícitamente y se define como:

$$\mathbf{h}(\mathbf{x}) = (\mathrm{g}_2(\mathbf{t}_l, \boldsymbol{x}), \forall l \in \{1, \ldots, L\}), \tag{10}$$

donde $\mathrm{g}_2(\mathbf{t}_l, \boldsymbol{x}) : \mathbb{R}^K \to \mathbb{R}$ es la función de activación del nodo oculto l-ésimo, $\mathbf{t}_l \in \mathbb{R}^{K+1}$ es el vector de pesos de entrada asociado al nodo oculto l-ésimo, y con $x_{K+1} = -1$.

Extreme Learning Machine

Función activación ELM neuronal

La función de activación suele ser típicamente la sigmoide, es decir:

$$\mathrm{g}_2(\mathbf{t}_l, \boldsymbol{x}) = g(\mathbf{t}_l'\mathbf{x}), \tag{11}$$

donde $g(t) = (1 + \exp(-t))^{-1}$.

Estimación parámetros de la función activación ELM neuronal

En la versión neuronal del marco de trabajo, los pesos de entrada de los nodos ocultos se eligen al azar, y la matriz de pesos de salida, $\mathbf{w}$, se determina analíticamente utilizando la Ec. 5 o la Ec. 6.

Flujo algorítmico

ELM-NEURONAL $(\mathcal{D},\ L,\ C)$:

$\mathbf{X} \leftarrow (\mathbf{X}\ \mathbf{1}_N) \in \mathbb{R}^{N\times(K+1)}$
$\mathbf{t} \leftarrow 2 \cdot \mathrm{rand}(L, K+1) - 1$
for $n = 1$ until N **do**
  for $l = 1$ until L **do**
    $\mathbf{H}_{nl} \leftarrow \mathrm{g}_2(\mathbf{t}_l, \mathbf{x}_n)$
  end for
end for
$\mathbf{w} \leftarrow \left(\frac{\mathbf{I}}{C} + \mathbf{H}'\mathbf{H}\right)^{-1} \mathbf{H}'\mathbf{Y}$
return $\mathbf{w}, \mathbf{t}$

Índice de contenidos

Extreme Learning Machine

Modelos kernel

En la implementación del ELM kernel, la función $\boldsymbol{h}(\mathbf{x})$ es un mapeo de características desconocido (a diferencia del modelo neuronal).

Kernel trick

Afortunadamente, existen ciertas funciones $k(\mathbf{x}_i, \mathbf{x}_j)$ que calculan el producto escalar en otro espacio, $k(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{h}(\mathbf{x}_i), \mathbf{h}(\mathbf{x}_j) \rangle$ (para todos los $\mathbf{x}_i$ y $\mathbf{x}_j$ en el espacio de entrada). Por lo tanto, reformularemos la solución para agrupar los términos de $\mathbf{h}(\mathbf{x})$ en productos escalares y aplicaremos el truco del kernel a ellos (sustituyendo esos elementos por sus funciones de kernel).

Extreme Learning Machine

La función de salida del clasificador ELM (después de aplicar el truco del kernel) para un patrón de prueba, $\mathbf{x}$, puede expresarse de manera compacta como:

$$\begin{aligned}\hat{\mathbf{y}}(\mathbf{x}) &= \mathbf{h}(\mathbf{x})\beta \\ &= \mathbf{h}(\mathbf{x})\mathbf{H}' \left(\frac{\mathbf{I}}{C} + \mathbf{H}\mathbf{H}'\right)^{-1} \mathbf{Y} \\ &= \mathbf{K}(\mathbf{x})' \left(\frac{\mathbf{I}}{C} + \mathbf{\Omega}_{\mathrm{ELM}}\right)^{-1} \mathbf{Y},\end{aligned} \tag{12}$$

donde $\mathbf{K} : \mathbb{R}^K \to \mathbb{R}^N$ es la función kernel vectorial

$$\mathbf{K}(\mathbf{x}) = (k(\mathbf{x}, \mathbf{x}_n), \forall n \in \{1, \ldots, N\} \in \mathbb{R}^N,$$

que nos señala la similitud del patrón que estamos testeando con respecto a los patrones del conjunto de entrenamiento

Extreme Learning Machine

La función de kernel más utilizada es la gaussiana:

$$k(\mathbf{u}, \mathbf{v}) = \exp(-\sigma||\mathbf{u} - \mathbf{v}||^2), \tag{13}$$

donde $\sigma \in \mathbb{R}$ es el parámetro del kernel (define el ancho de la gaussiana). La matriz de kernel $\mathbf{\Omega}_{\text{ELM}} = (\Omega_{i,j})_{i,j=1,\ldots,n}$ se define como:

$$\Omega_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j). \tag{14}$$

Flujo algorítmico

ELM-KERNEL-TRAIN $(\mathcal{D}, \sigma, C)$:

$\mathbf{t} \leftarrow 2 \cdot \mathrm{rand}(L, K+1) - 1$
for $i = 1$ until N **do**
  for $j = 1$ until N **do**
    $\Omega_{i,j} \leftarrow \exp(-\sigma||\mathbf{x}_i - \mathbf{x}_j||^2)$
  end for
end for
$\mathbf{w} \leftarrow \left(\frac{\mathbf{I}}{C} + \boldsymbol{\Omega}_{\mathrm{ELM}}\right)^{-1} \mathbf{Y}$
return $\mathbf{w}$

ELM-KERNEL-TEST $(\mathbf{x})$:

$\hat{\mathbf{y}}(\mathbf{x}) \leftarrow \mathbf{K}(\mathbf{x})'\mathbf{w}$
return $\hat{\mathbf{y}}(\mathbf{x})$

Índice de contenidos

Regla de clasificación

- La función de salida de ELM $\hat{\mathbf{y}}(\mathbf{x}_n) = \mathbf{h}'(\mathbf{x}_n)\,\mathbf{w}$ da como resultado la salida del patrón $\mathbf{x}_n$ en formato continuo, es decir, $\hat{\mathbf{y}}(\mathbf{x}_n) \in \mathbb{R}^J$.
- En el problema de clasificación, se requieren salidas en formato discreto, es decir, $\{0, 1\}^J$.
- Para lograr esto, se realiza una discretización de la salida continua de la red, creando un vector $\hat{\mathbf{o}}(\mathbf{x}_n) \in \{0, 1\}^J$ cuya componente j-ésima es igual a 1 si la dimensión correspondiente en $\hat{\mathbf{y}}(\mathbf{x}_n)$ es la máxima, y cero en caso contrario.

¡Gracias por vuestra atención!