Redes de Hopfield II

Modelos de la computación (Redes Recurrentes y Autónomas)

Francisco Fernández Navarro

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

Índice de contenidos

Introducción a la Optimización

- **Objetivo de Optimización**: Encontrar la mejor solución en un conjunto de posibles soluciones bajo ciertas restricciones.
- Aplicaciones de la Optimización: Economía, ingeniería, logística, planificación, etc.
- Tipos de Optimización: Lineal, no lineal, combinatoria, etc.

Problema de la Mochila (Knapsack)

- **Definición del Problema**: Seleccionar elementos para una mochila maximizando el valor total sin exceder la capacidad de peso.
- Variables de Optimización:

$$x_i \in \{0, 1\} : \text{Indica si el elemento } i \text{ se coloca en la mochila}$$

- Función Objetivo:

$$\text{Maximizar } \sum_{i=1}^{n} v_i \cdot x_i$$

donde v_i es el valor del elemento i.
- Restricciones:

$$\sum_{i=1}^{n} w_i \cdot x_i \leq C$$

donde w_i es el peso del elemento i y C es la capacidad de la mochila.

Ejemplo del Problema de la Mochila

Supongamos que tenemos una mochila con capacidad $C = 15$ unidades de peso y los siguientes elementos:

- Elemento 1: $v_1 = 10$, $w_1 = 2$
- Elemento 2: $v_2 = 6$, $w_2 = 5$
- Elemento 3: $v_3 = 12$, $w_3 = 9$
- Elemento 4: $v_4 = 8$, $w_4 = 5$

Formulemos el problema de la mochila con estos datos.

Resolución del Ejemplo

- Variables de Optimización: x_1, x_2, x_3, x_4
- Función Objetivo:

$$\text{Maximizar } 10x_1 + 6x_2 + 12x_3 + 8x_4$$

- Restricción de Capacidad:

$$2x_1 + 5x_2 + 9x_3 + 5x_4 \leq 15$$

Programación lineal de enteros

Presentación en formato matricial

En formato matricial el problema quedaría definido como:

$$\begin{aligned} \max_{\mathbf{x}\in\{0,1\}^N} \quad & \mathbf{v}'\mathbf{x}. \\ \text{s.t.} \quad & \mathbf{w}'\mathbf{x} \leq C, \end{aligned} \tag{1}$$

donde $\mathbf{x} \in \{0,1\}^N$ es el vector binario a determinar, $\mathbf{v}, \mathbf{w} \in \mathbb{R}^N$ son los vectores con los valores y los pesos de los N objetos considerados en el problema, y finalmente, C la capacidad total de la mochila.

Resolución en MATLAB

En MATLAB podríamos resolverlo con la rutina intlinprog.

Resolución mediante Redes de Hopfield

Objetivo

Adaptar la función de energía del Modelo de Hopfield a otro problema de optimización, para encontrar su solución utilizando el hecho de que la dinámica conduce a un mínimo.

Cuidado

Puede pasar que la solución encontrada no sea muy buena (Mínimo local con una energía alta)

Índice de contenidos

Definición del problema

Definición del problema

Aproximar una señal continua (analógica) $y \in [0, 3]$ a través de la ecuación $\hat{y} = x_0 + 2x_1$, donde $x_0, x_1 \in [0, 1]$.

Definición de restricciones

Además debemos asegurarnos que:

- $x_0^2 = x_0 \Rightarrow x_0^2 - x_0 = x_0(1 - x_0) = 0$
- $x_1^2 = x_1 \Rightarrow x_1^2 - x_1 = x_1(1 - x_1) = 0$

Formulación del problema de optimización

El problema de optimización quedaría definido como:

$$
\begin{aligned}
\min_{\mathbf{x}\in[0,1]^2} \quad & (y-(x_0+2x_1))^2. \\
\text{s.t.} \quad & x_0(1-x_0)=0 \\
& x_1(1-x_1)=0.
\end{aligned} \tag{2}
$$

El Lagrangiano del problema de optimización:

$$
\mathcal{L}(\mathbf{x},\boldsymbol{\lambda}) = \underbrace{(y-(x_0+2x_1))^2}_{\text{Función Objetivo}} + \underbrace{\lambda_0(x_0(1-x_0)) + \lambda_1(x_1(1-x_1))}_{\text{Restricciones}}, \tag{3}
$$

donde $\lambda_0, \lambda_1 \in \mathbb{R}$ representan a los multiplicadores de Lagrange.

Formulación del problema de optimización

Desarrollando el Lagrangiano:

$$
\begin{aligned}
\mathcal{L}(\mathbf{x},\boldsymbol{\lambda}) = {} & \underbrace{(y-(x_0+2x_1))^2}_{\text{Desarrollamos este término}} + \lambda_0(x_0(1-x_0)) + \lambda_1(x_1(1-x_1)) \\
&= y^2 - 2y(x_0+2x_1) + \underbrace{(x_0+2x_1)^2} + \lambda_0(x_0(1-x_0)) + \lambda_1(x_1(1-x_1)) \\
&= \underbrace{y^2 - 2yx_0 - 4yx_1 + x_0^2 + 4x_0x_1 + 4x_1^2 + \lambda_0(x_0(1-x_0)) + \lambda_1(x_1(1-x_1))}_{\text{Agrupamos los términos}} \\
&= y^2 + x_0^2(1-\lambda_0) + x_1^2(4-\lambda_1) + x_0(\lambda_0-2y) + x_1(\lambda_1-4y) + 4x_0x_1
\end{aligned}
$$

En el modelo de Hopfield no existen términos cuadráticos (no se consideran las auto-conexiones), por lo que para poder anularlos debemos fijar $\lambda_0 = 1$ y $\lambda_1 = 4$. Con estos valores, y eliminando y^2 ya que no depende de x_0 ni x_1, la función de energia quedaría como:

$$E = (1-2y)x_0 + 4(1-y)x_1 + 4x_0x_1$$

Formulación del problema de optimización

Reescribimos la función de energía para que tenga la forma

$$E = -\frac{1}{2}w_{01}x_0x_1 - \frac{1}{2}w_{10}x_1x_0 + \theta_0x_0 + \theta_1x_1, \qquad (4)$$

por lo que

$$E = \underbrace{-\frac{1}{2}(-4x_0x_1 - 4x_1x_0)}_{w_{01},w_{10}} + \underbrace{(1-2y)x_0}_{\theta_0} + \underbrace{4(1-y)x_1}_{\theta_1},$$

y consecuentemente

$$w_{01} = w_{10} = -4, \theta_0 = (1-2y), \theta_1 = 4(1-y).$$

Índice de contenidos

Problema de las Torres en el Ajedrez

Objetivo del Problema

Situar N torres en un tablero de $N \times N$ sin que ninguna torre pueda atacar a otra.

			T
	T		
		T	
T			

En el tablero de ajedrez de 4×4, hemos situado 4 torres marcadas como "T". El objetivo es que ninguna torre pueda atacar a otra.

Resolución del problema

Variables de optimización

$\mathbf{s} \in \{0,1\}^{N\times N}$: Tablero de ajedrez de tamaño $N \times N$, siendo N el número de torres a incluir. Cada elemento de la matriz ($i, j \in \{1, \ldots, N\}$):

$$s_{ij} = \begin{cases} 1 & \text{si hay una torre en la fila } i\text{, columna } j \\ 0 & \text{en caso contrario} \end{cases}$$

Restricción 1: En cada fila solo podemos situar a una torre

$$\sum_{j=1}^{N} s_{ij} = 1, \quad \forall i \in \{1, \ldots, N\} \quad \begin{cases} s_{11} + s_{12} + \ldots + s_{1N} = 1 \\ s_{21} + s_{22} + \ldots + s_{2N} = 1 \\ \vdots \\ s_{N1} + s_{N2} + \ldots + s_{NN} = 1 \end{cases}$$

Resolución del problema

Restricción 1: En cada fila solo podemos situar a una torre

$$\sum_{i=1}^{N} s_{ij} = 1, \quad \forall j \in \{1, \ldots, N\} \quad \begin{cases} s_{11} + s_{21} + \ldots + s_{N1} = 1 \\ s_{12} + s_{22} + \ldots + s_{N2} = 1 \\ \vdots \\ s_{1N} + s_{1N} + \ldots + s_{NN} = 1 \end{cases}$$

Función de coste

Sólo tenemos restricciones, no hay función a minimizar o maximizar.

Resolución del problema

La función de energía queda como (las restricciones al cuadrado para que los errores por la izquierda no compensen los de la derecha):

$$E = \sum_{i=1}^{N} \left(\sum_{j=1}^{N} s_{ij} - 1 \right)^2 + \sum_{j=1}^{N} \left(\sum_{i=1}^{N} s_{ij} - 1 \right)^2$$

Desarrollamos el primer término:

$$\begin{aligned}
\sum_{i=1}^{N} \left(\sum_{j=1}^{N} s_{ij} - 1 \right)^2 &= \sum_{i=1}^{N} \left[\left(\sum_{j=1}^{N} s_{ij} \right) \left(\sum_{k=1}^{N} s_{ik} \right) + 1 - 2 \sum_{j=1}^{N} s_{ij} \right] \\
&= \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{N} s_{ij} s_{ik} + N - 2 \sum_{i=1}^{N} \sum_{j=1}^{N} s_{ij} \\
&= \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1, k \neq j}^{N} s_{ij} s_{ik} + \sum_{i=1}^{N} \sum_{j=1}^{N} s_{ij}^2 + N - 2 \sum_{i=1}^{N} \sum_{j=1}^{N} s_{ij}
\end{aligned}$$

Resolución del problema

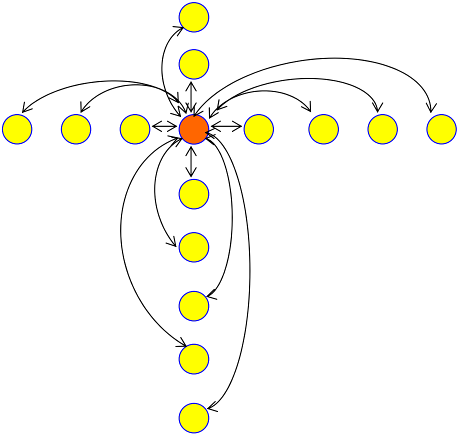
Si tenemos en cuenta que $s_{ij}^2 = s_{ij}$ (por la naturaleza de la variable):

$$\begin{aligned}
\sum_{i=1}^{N}\left(\sum_{j=1}^{N} s_{ij} - 1\right)^2 &= \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1,k\neq j}^{N} s_{ij}s_{ik} + \underbrace{\sum_{i=1}^{N}\sum_{j=1}^{N} s_{ij}^2 - 2\sum_{i=1}^{N}\sum_{j=1}^{N} s_{ij}}_{-\sum_{i=1}^{N}\sum_{j=1}^{N} s_{ij}} + N \\
&= \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1,k\neq j}^{N} s_{ij}s_{ik} - \sum_{i=1}^{N}\sum_{j=1}^{N} s_{ij} + N
\end{aligned}$$

De esta forma la función de energía quedaría como:

$$\begin{aligned}
E &= \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1,k\neq j}^{N} s_{ij}s_{ik} - \sum_{i=1}^{N}\sum_{j=1}^{N} s_{ij} + \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{r=1,r\neq i}^{N} s_{ij}s_{rj} - \sum_{i=1}^{N}\sum_{j=1}^{N} s_{ij} + 2N \\
&= -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1,k\neq j}^{N} (-2)s_{ij}s_{ik} - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{r=1,r\neq i}^{N} (-2)s_{ij}s_{rj} + \sum_{i=1}^{N}\sum_{j=1}^{N} (-2)s_{ij}
\end{aligned}$$

Solución al problema



$$w_{ij,ik} = -2 \text{ cuando } k \neq j$$
$$w_{ij,rj} = -2 \text{ cuando } r \neq i$$
$$w_{ij,ij} = 0$$
$$w_{ij,rk} = 0$$
$$\theta_{ij} = -2$$

Cada neurona está conectada con otras neuronas de su misma fila y neuronas de su misma columna.

Cuidado con la solución...

A tener en cuenta

El problema de las N Torres en realidad requiere resolver dos tipos de problema: por fila y por columna. Cuando estamos sumando los valores del sesgo, θ_{ij}, en realidad es como si dijeramos que tenemos un esquema OR en lugar de un esquema AND. Por ello debemos dejar el valor de θ_{ij} a -1.

Índice de contenidos

Definición del problema

Definición del problema

Supongamos una ciudad representada en un grafo en el que las aristas del grafo son las calles y los vértices los puntos de concurrencia. El objetivo sería el de vigilar todas las calles con cámaras de vídeo desde los puntos de concurrencia, instalando en dichos puntos el menor número de cámaras (reducir coste).

Resolución del problema

Variables de optimización

$\mathbf{s} \in \{0,1\}^N$: Vector con los N vértices del grafo. Cada elemento del vector ($i \in \{1, \ldots, N\}$):

$$s_i = \begin{cases} 1 & \text{si el vértice } i \text{ es seleccionado} \\ 0 & \text{en caso contrario} \end{cases}$$

Restricción 1: Si hay calle entre vértice i y j, ésta debe estar vigilada

$$\sum_{i=1}^{N} \sum_{j=1, j\neq i}^{N} a_{ij} \underbrace{(1-s_i)(1-s_j)}_{\text{Al menos una debe estar activa}} = 0,$$

donde $a_{ij} \in \{0,1\}$ indica si hay calle entre los vértices i y j.

Resolución del problema

Función objetivo: Minimizar el número de cámaras

$$\sum_{i=1}^{N} s_i$$

La función de energía incorporando la función objetivo y la restricción con su parámetro correspondiente quedaría como:

$$E = \sum_{i=1}^{N} s_i + \lambda \sum_{i=1}^{N} \sum_{j=1, j \neq i}^{N} a_{ij}(1 - s_i)(1 - s_j),$$

donde $\lambda \in \mathbb{R}$ es el multiplicador de Lagrange.

Resolución del problema

Desarrollando la función de energía nos queda que:

$$\begin{aligned} E &= \sum_{i=1}^{N} s_i \underbrace{+\lambda \sum_{i=1}^{N} \sum_{j=1, j\neq i}^{N} a_{ij}(1-s_i)(1-s_j)}_{\text{Desarrollamos este término}} \\ &= \sum_{i=1}^{N} s_i + \lambda \sum_{i=1}^{N} \sum_{j=1, j\neq i}^{N} a_{ij} + \lambda \sum_{i=1}^{N} \sum_{j=1, j\neq i}^{N} a_{ij} s_i s_j \\ & \underbrace{-\lambda \sum_{i=1}^{N} \sum_{j=1, j\neq i}^{N} a_{ij} s_i - \lambda \sum_{i=1}^{N} \sum_{j=1, j\neq i}^{N} a_{ij} s_j}_{-2\lambda \sum_{i=1}^{N} \sum_{j=1, j\neq i}^{N} a_{ij} s_j} . \end{aligned}$$

Resolución del problema

Reescribimos la función para que el problema pueda ser resuelto por una red de Hopfield:

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1,j\neq i}^{N}(-2\lambda)a_{ij}s_i s_j + \sum_{i=1}^{N}\left(1 - 2\lambda\sum_{j=1,j\neq i}^{N}a_{ij}\right)s_i.$$

De esta forma tenemos que:

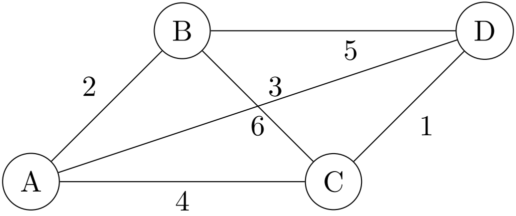
$$\begin{aligned} w_{ij} &= -2\lambda a_{ij} \text{ cuando } i \neq j \\ w_{ii} &= 0 \\ \theta_i &= \left(1 - 2\lambda\sum_{j=1,j\neq i}^{N}a_{ij}\right) \end{aligned}$$

Índice de contenidos

Problema del Viajante de Comercio (TSP)

Definición

El Problema del Viajante de Comercio (TSP) es un problema de optimización combinatoria que busca encontrar el recorrido más corto que visita un conjunto de ciudades y regresa al punto de partida, minimizando la distancia total recorrida.



En el ejemplo anterior, se ilustra el TSP con cuatro ciudades y las distancias entre ellas.

Resolución del problema

Variables de optimización

$\mathbf{s} \in \{0,1\}^{N\times N}$: Matriz de tamaño $N \times N$, siendo N el número de ciudades, que indica en su elemento i,j, s_{ij} si la ciudad i se visita el día j (1 si es así y 0 en caso contrario).

Restricción 1

Cada ciudad debe ser visitada un único día:
$\sum_{j=1}^{N} s_{ij} = 1, i \in \{1, \ldots, N\}$.

Restricción 1

Además, sólo una ciudad se puede visitar en la posición j (tramo j-ésimo de la ruta): $\sum_{i=1}^{N} s_{ij} = 1, j \in \{1, \ldots, N\}$.

Resolución del problema

Función objetivo: Minimizar la longitud total de la ruta

- Necesitamos una matriz de distancias $\mathbf{d} \in \mathbb{R}^{N \times N}$ que indique la distancia de la ciudad i a la j.
- Las ciudades i y j estarán conectadas si $s_{ik} = 1$ y $s_{jk\pm1} = 1$ en alguno de los N pasos de la ruta, $k \in \{1, \ldots, N\}$.
- Para cada par de ciudades i, j, tendremos un coste asociado de $\sum_{k=1}^{N} d_{ij} s_{ik}(s_{j,k-1} - s_{j,k+1})$.

Por todo ello, la función objetivo quedará definida como:

$$\frac{1}{2} \sum_{k=1}^{N} \sum_{i=1}^{N} \sum_{j=1, j\neq i}^{N} d_{ij} s_{ik}(s_{j,k-1} - s_{j,k+1})$$

Se divide por dos porque una misma ruta aparece dos veces en la función objetivo, una en cada dirección.

Formulación de la función de energía

En base a la función de costo y las restricciones mencionadas anteriormente, la función de energía final E puede ser expresada de la siguiente manera:

$$
\begin{aligned}
E &= \frac{1}{2}\sum_{k=1}^{N}\sum_{i=1}^{N}\sum_{j=1,j\neq i}^{N} d_{ij}s_{ik}(s_{j,k-1}-s_{j,k+1}) \\
&\quad +\lambda_0\sum_{i=1}^{N}\left(1-\sum_{j=1}^{N}s_{ij}\right)^2+\lambda_1\sum_{j=1}^{N}\left(1-\sum_{i=1}^{N}s_{ij}\right)^2
\end{aligned}
$$

donde $\lambda_0, \lambda_1 \in \mathbb{R}$ son los multiplicadores de Lagrange. Las restricciones se incluyen al cuadrado para evitar que errores en negativo compensen errores en positivo.

¡Gracias por vuestra atención!