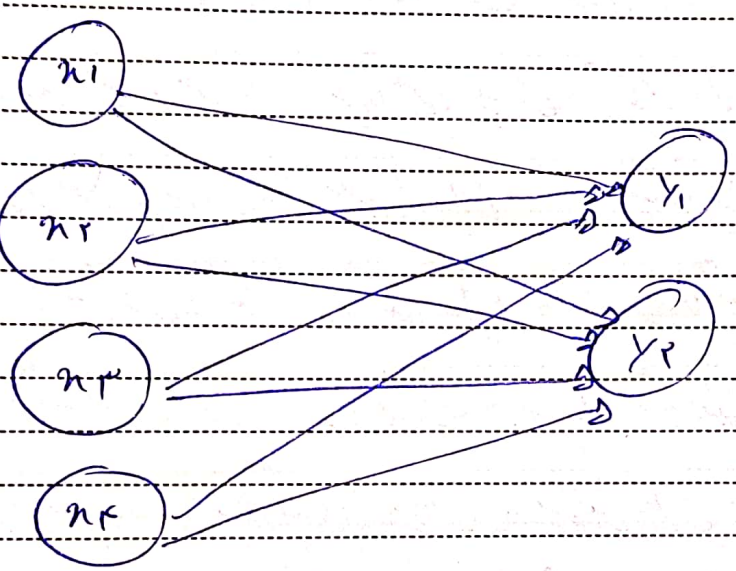
امیر محمد کمیجانی ۹۹۵۲۲۰۳۲



$$w_1 = [0.2, 0.4, 0.6, 0.8] \quad , \quad w_2 = [0.3, 0.5, 0.7, 0.9]$$

$$x_1 = [1\ 1\ 0\ 0] \quad x_2 = [0\ 0\ 0\ 1]$$

$$x_3 = [0\ 1\ 1\ 0] \quad x_4 = [1\ 0\ 0\ 0]$$

به طور کلی ابتدا x_i ها را در w_j ها ضرب می‌کنیم و حاصل آن‌ها را با هم جمع می‌کنیم هر جا که حاصل آن‌ها بیشتر بود آن نورون خروجی برنده است و سپس وزن‌ها را آپدیت می‌کنیم و به همین ترتیب ادامه می‌دهیم.

Scanned with CamScanner

نقطه اول:

$x_1 = [1\ 1\ 0\ 0]$

$\arg\min \|x - w_j\|$

d_1^2 = فاصله اقلیدسی برای نورون خروجی اول

$d_1^2 = (0.2-1)^2 + (0.4-1)^2 + (0.6)^2 + (0.8)^2$

$d_1^2 = 2$

$d_2^2 = (0.9-1)^2 + (0.7-1)^2 + (0.5)^2 + (0.3)^2 = 0.44$

به ازای $d_2^2 < d_1^2$ (فقط مقادیر مثبت اهمیت) $\Rightarrow$ نورون خروجی ۲ یا y_2 برنده است

آپدیت وزن اول: چون گفتیم تابع همسایگی در نظر نگیریم وزن ها را با فرمول زیر آپدیت می کنیم

در اصل فرض کردیم $h_{j,i} = 1$

$$w_j(t+1) = w_j(t) + \eta(t)\,(x_s - w_j(t))$$

وزن آپدیت شده برای نورون y_2 که برنده بود بدین صورت می شود:

$$w_2(t+1) = [0.9\ 0.7\ 0.5\ 0.3] + 0.5\,(\underbrace{[1\ 1\ 0\ 0] - [0.9\ 0.7\ 0.5\ 0.3]}_{[0.1\ \ 0.1\ \ -0.5\ \ -0.3]})$$

$$w_2(t+1) = [1.4\ \ 1.2\ \ 0.25\ \ 0.15]$$

$$w_1 = [0.2\ 0.4\ 0.6\ 0.8]$$

تصویر سوم:

$x_2 = [0 \quad 0 \quad 0 \quad 1]$

$w_1 = [0/2 \quad 0/4 \quad 0/6 \quad 0/8] \quad w_2 = [1/4 \quad 1/2 \quad 0/25 \quad 0/18]$

همانند قبل فاصله اقلیدسی را محاسبه می‌کنیم:

$d_1^2 = (0/2)^2 + (0/4)^2 + (0/6)^2 + (0/8-1)^2 = 0/68$

$d_2^2 = (1/4)^2 + (1/2)^2 + (0/25)^2 + (0/18-1)^2 = 4/185$

مشخص است نورون خروجی ۱ برنده می‌شود

آپدیت وزن:

$w_1(t+1) = [0/2 \quad 0/4 \quad 0/6 \quad 0/8] + 0/5\big([0 \; 0 \; 0 \; 1] - [0/2 \; 0/4 \; 0/6 \; 0/8]\big)$

$w_1(t+1) = [0/1 \quad 0/2 \quad 0/3 \quad 0/7]$

$w_2 = [1/4 \quad 1/2 \quad 0/25 \quad 0/18]$

★ همانطور که در ابتدا توضیح دادم و در تمام مراحل هم مشخص است ابتدا فاصله اقلیدسی با توجه به وزن‌ها و ورودی را محاسبه کنیم و فاصله اقلیدسی که کمتر است را به عنوان نورون برنده معرفی می‌کنیم سپس آپدیت وزن را برای نورون برنده انجام می‌دهیم. برای نقاط ۳ و ۴ به همین ترتیب ادامه می‌دهیم.

حقیقت اول (۲)

$$X_1: \begin{bmatrix}1\\1\\1\\1\end{bmatrix}\begin{bmatrix}1&1&1&1\end{bmatrix}=\begin{bmatrix}1&1&1&1\\1&1&1&1\\1&1&1&1\\1&1&1&1\end{bmatrix}$$

$$X_2: \begin{bmatrix}-1\\-1\\-1\\-1\end{bmatrix}\begin{bmatrix}-1&-1&-1&-1\end{bmatrix}=\begin{bmatrix}1&1&1&1\\1&1&1&1\\1&1&1&1\\1&1&1&1\end{bmatrix}$$

$$X_3: \begin{bmatrix}-1\\-1\\1\\1\end{bmatrix}\begin{bmatrix}-1&-1&1&1\end{bmatrix}=\begin{bmatrix}1&1&-1&-1\\1&1&-1&-1\\-1&-1&1&1\\-1&-1&1&1\end{bmatrix}$$

$$\Rightarrow X_1+X_2+X_3+X_4=\begin{bmatrix}4&4&0&0\\4&4&0&0\\0&0&4&4\\0&0&4&4\end{bmatrix}$$

← حول از ها نمایید استفاده می‌کنیم می‌توانیم بگوییم تمام مقادیر متعامد اصلی برابر صفر هستند (در بخش بعدی اثبات داریم)

Scanned with CamScanner

۹ شوال ۱۴۴۴ 2023 Apr 30

ادامه سوال (۲) قسمت اول)

$$E = -\frac{1}{2}\sum_{i=1}^{p}\sum_{j=1}^{p} w_{ij} n_i n_j = -\frac{1}{2}\left(\frac{4+4+\cdots}{24}\right) = -16$$

8:00

9:00

مقدار E منفی است. چون مقدار انرژی کمتر باشد قابلیت stable بودن آن بیشتر است. میدانیم در شبکه هاپفیلد هنگامی به وضعیت بهتری می‌رویم انرژی کاهش می‌یابد همچنین شبکه به وضعیتی رسیده است که با Pattern های ذخیره شده مطابقت دارد.

10:00

11:00

⟸ در کل می‌توان گفت قابلیت ذخیره‌سازی می‌یابد.

$w_{ij} = \{ x_i^k \; x_j^k$ متمم (۲)

Weight matrice :

$$\begin{bmatrix} 0 & 4 & 0 & 0 \\ 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 \\ 0 & 0 & 4 & 0 \end{bmatrix}$$

$$E = -\frac{1}{2} \sum_{i=1}^{p} \sum_{j=1}^{p} w_{ij} x_i x_j$$

ورودی $[1\,1\;\;1\,1]$

جدول تابعیت برای ورودی ۱:

	۱	۲	۳	۴
t_1 = input	1	1	1	1
t_2	1	1	1	1
t_3	1	1	1	1
t_n	1	1	1	1
$\sum x_i w_{ij}$	۴	۴	۴	۴
	۴	۴	۴	۴

⇐ stable است

input = [-1, -1, -1, -1]

	١	٢	٣	٤
t_1 = input	-1	-1	-1	-1
t_2	-1	-1	-1	-1
t_n	-1	-1	-1	-1
$\sum x_i w_{ij}$	-٤	-٤	-٤	-٤

=) stable

input = [-1, -1, 1, 1]

	١	٢	٣	٤
input = t_1	-1	-1	1	1
t_2	-1	-1	1	1
t_n	-1	-1	1	1
$\sum x_i w_{ij}$	-٤	-٤	٤	٤

=) stable

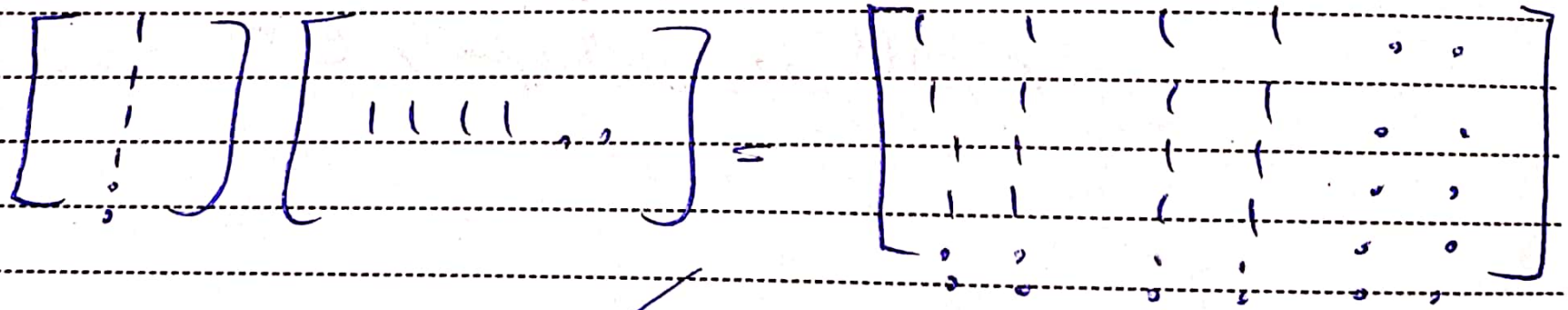
وزایمنی حمل و نقل

input = [1, 1, -1, -1]

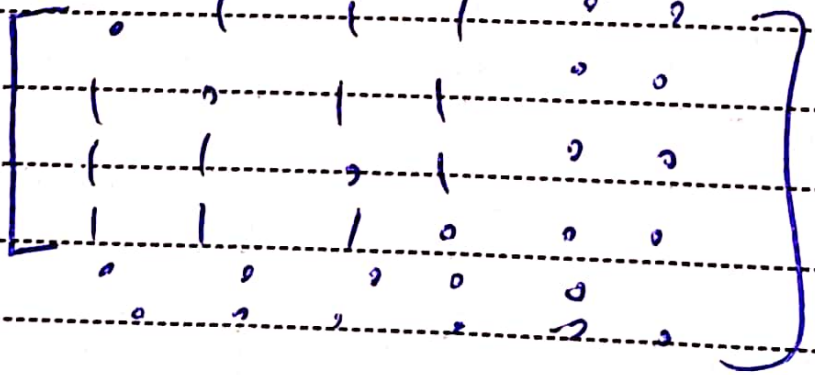
	١	٢	٣	٤
t_1 = input	1	1	-1	-1
t_2	1	1	-1	-1
t_n	1	1	-1	-1
$\sum x_i w_{ij}$	٤	٤	-٤	-٤

<=> stable

weight matrice:



مقادیر مقدار اصلی را ضرب می‌کنیم



threshold = 0

$\begin{cases} 1 & n \geq 0 \\ 0 & n < 0 \end{cases}$

جدول محاسبه را رسم می‌کنیم:

	۱	۲	۳	۴	۵	۶
input = t_1	۰	۱	۰	۰	۰	۰
$t=۲$	۱	۰	۱	۱	۰	۰
$t=۳$	۱	۱	۱	۱	۰	۰
$t=۴$	۱	۱	۱	۱	۰	۰
$\{\lambda_i, w_i\}$	۱	۰	۱	۱	۰	۰
	۲	۳	۲	۲	۰	۰
	۳	۳	۳	۳	۶	۰

8:00
9:00
10:00
11:00
12:00
13:00
14:00

$\}=$ stable

$t=۳$ بوا سیستم به الگوی ۰۱۱۱۱۰ میرسیم

۱۲
سه‌شنبه اردیبهشت
2023 May 2 ۱۱ شوال ۱۴۴۴

TSP: Travelling salesman Problem

۵) بررسی SOM / Kohonen

مدل کوهونن معمولا برای مسائل clustering استفاده می‌شوند.
در این مسئله به طور مثال می‌توانیم شهرها را براساس فاصله‌ای که دارند در map کوهونن طبقه‌بندی کنیم اما برای اینکه کمترین هزینه برای طی کردن تمام شهرها را به دست آوریم کمی نامناسب و نامعمول است. (می‌توانیم وزن‌ها را فاصله در نظر بگیریم)
همچنین می‌دانیم TSP یک نوع مسئله optimization می‌باشد
مدلی نیست که برای این کار مناسب باشد. Kohonen
← دنبال رابطه‌ای بین دیتاها هستیم تا نزدیک‌ترین آنها را به هم همسایه اعلام کنیم
همچنین TSP محدودیت هایی دارد مانند اینکه از یک شهر باید یکبار گذر کرد که
تبدیل آن در SOM سخت می‌باشد

بررسی MLP

این مدل یک مدل supervised می‌باشد یعنی باید دیتایی برای train آن تهیه کنیم. از مزیت این مدل می‌توان گفت که برای مسائل regression و approximation مناسب است. این مدل می‌تواند محدودیت دیگری مثل محدودیت تعداد نورون داشته باشد که اگر تعداد شهرها در TSP افزایش یابد با چالش روبه‌رو می‌شویم. اما خاصیت global min در MLP می‌تواند مفید باشد اما باز هم این مدل برای مسائل optimize خوب نیست

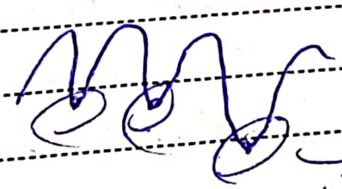
بررسی Hopfield

این مسئله با هاپفیلد قابل حل می باشد چون بعد از اینکه برای بار اول ورودی دادیم را حل
شبکه را دائما در حال iterate کردن هستیم تا به یک وضعیت stable برسیم یعنی بتواند به ما
بهترین و most optimized جواب را بدهد.

می توانیم به هر شهر یک نورون اختصاص دهیم البته باز هم اگر تعداد شهرها زیاد شود می تواند
مشکل ساز شود، همچنین وزن بین نورون های فاصله شهرها

می توانیم هنگامی که تمام شرایط TSP اعمال شد یک سیکل تمام شهرها و با کاهش فاصله بین
شهرها را برابر با stable شدن در hopfield بدانیم
همچنین وزن ها را در هر iteration دائما آپدیت می کنیم

هر کل خاصیت memory داشتن هاپفیلد هم می تواند بسیار کمک کننده باشد
البته اینکه هاپفیلد همواره جواب بهینه را پیدا کند ضمانت نمی شود مثلا در شکل های
جزوه داریم:



تمام این نقاط stable هستند

اما لزوما global min سیستم برضمانت شود