Liet. matem. rink., T. 43, spec. nr., 218–222

Lygiagretūs skaičiavimai savireguliuojančio neuroninio tinklo junginyje su Sammono algoritmu

Gintautas DZEMYDA, Olga KURASOVA, Virginijus MARCINKEVIČIUS (MII)
el. paštas: dzemyda@ktl.mii.lt, kurasova@ktl.mii.lt, virgism@ktl.mii.lt

1. Įvadas

Savireguliuojančio neuroninio tinklo (SOM) [5] junginys su Sammono algoritmu [7] yra vienas iš daugiamačių duomenų vizualizavimo (projektavimo į plokštumą) būdų. Daugiamačiai duomenys yra analizuojami Sammono algoritmu įvertinant SOM mokymo rezultatus arba net ir mokymosi eigą. Darbuose [3, 4] pateiktas tokio tipo nuoseklusis algoritmas, įvertinantis SOM mokymosi eigą. Šio algoritmo skaičiavimai reikalauja nemažai laiko. Lygiagretieji algoritmai yra vienas iš būdų ilgai trunkantiems skaičiavimams pagreitinti. Kad lygiagretusis algoritmas būtų efektyvesnis, būtina ištirti ir suderinti jo realizacijoje eilę svarbių faktorių. Atlikta analizė leido parinkti optimalius (a) SOM mokymo epochų, (b) SOM mokymo blokų bei (c) Sammono algoritmo iteracijų skaičius.

2. SOM ir Sammono algoritmai

Savireguliuojantis neuroninis tinklas (SOM) yra neuronų m_{ij} ($i = 1, \ldots, k_x$, $j = 1, \ldots, k_y$), išdėstytų dvimačio tinklelio (lentelės) mazguose, masyvas. Keturkampės tinklo struktūros atveju, k_x yra lentelės eilučių skaičius, k_y – stulpelių skaičius. Kiekvienas n-matis apmokymo aibės vektorius $X \in (X_1, X_2, \ldots, X_s)$ mokymo metu yra susiejamas su vienu tinklo neuronu, kuris taip pat yra n-matis vektorius. Mokymo pradžioje vektorių m_{ij} komponentės generuojamos atsitiktinai. Kiekviename mokymo žingsnyje vienas iš apmokymo aibės vektorių $X \in (X_1, X_2, \ldots, X_s)$ pateikiamas į tinklą. Randama, iki kurio neurono m_c vektoriaus X Euklidinis atstumas yra mažiausias. Vektorius m_c pavadinamas neuronu-nugalėtoju. Neuronų komponentės keičiamos pagal (1) formulę

$$m_{ij} \leftarrow m_{ij} + h_{ij}^c (X - m_{ij}). \tag{1}$$

Čia, $h_{ij}^c = \frac{\alpha}{\alpha \eta_{ij}^c + 1}$, $\alpha = \max\left(\frac{e+1-\hat{e}}{e}, 0.01\right)$ (e – mokymo epochų skaičius, $\hat{e}$ – vykdomos epochos numeris, viena mokymo epocha – tai mokymo proceso dalis, kai visus vektorius pateikiame tinklui po vieną kartą, ją sudaro s mokymo žingsnių). Dydis η_{ij}^c yra kaimynystės tarp neuronų m_c ir m_{ij} eilė. Greta neurono-nugalėtojo esantys neuronai vadinami pirmos eilės kaimynai, greta pirmos eilės kaimynų esantys neuronai, išskyrus jau paminėtus – antros eilės kaimynai ir t.t. [2]. Kiekvienos epochos metu perskaičiuojami tie neuronai m_{ij}, kuriems

$$\eta_{ij}^c \leqslant \max\left[\alpha \max(k_x, k_y), 1\right]. \tag{2}$$

Pažymėkime $k = \max(k_x, k_y)$, funkciją $\eta(\hat{e}) = \max[\alpha \max(k_x, k_y), 1] = \max[\alpha k, 1]$. Sveikas skaičius n' rodo, kiek sumažėjo kaimynystės eilė lyginant su didžiausia (pagal (2) formulę didžiausia kaimynystė eilė $\eta_{ij}^c = k$, kai $\hat{e} = 1$). Paprastai k_x ir k_y, o tuo pačiu ir k, neviršija dešimčių. Tada teisinga tokia teorema.

Teorema. *Epochose* $\hat{e} = \left[\frac{(n'-1)e}{k}\right] + 2$ *(*$n' = 1, \ldots, k-1$*) neurono* m_c *maksimali kaimynystės eilė* η_{ij}^c *yra mažesnė vienetu, lyginant su (*$\hat{e} - 1$*)-ma epocha, jei* $1 \leqslant \hat{e} \leqslant e + 1 - \frac{e}{k}$*. Maksimali kaimynystės eilė nebemažėja ir lieka lygi vienam (* $\eta_{ij}^c = 1$*), kai* $e + 1 - \frac{e}{k} \leqslant \hat{e} \leqslant e$*.*

Įrodymas. Nesunku įsitikinti, kad augant vykdomos epochos numeriui $\hat{e}$, dydis $\alpha = \max\left(\frac{e+1-\hat{e}}{e}, 0.01\right)$ mažėja. Jis pasiekia ribinį tašką, kai $\frac{e+1-\hat{e}}{e} = 0.01 \Rightarrow \hat{e} = 0.99e+1$. Tada

$$\alpha = \max\left(\frac{e+1-\hat{e}}{e}, 0.01\right) = \begin{cases} \dfrac{e+1-\hat{e}}{e}, & \text{kai } 1 \leqslant \hat{e} \leqslant 0.99e+1, \\ 0.01, & \text{kai } 0.99e+1 < \hat{e} \leqslant e. \end{cases}$$

Augant vykdomos epochos numeriui $\hat{e}$, funkcija $\eta(\hat{e})$ mažėja. Ji pasiekia minimumą, kai $\alpha k = 1 \Rightarrow \frac{e+1-\hat{e}}{e} k = 1 \Rightarrow \hat{e} = e + 1 - \frac{e}{k}$. Tada,

$$\text{kai } 1 \leqslant \hat{e} \leqslant 0.99e+1, \quad \eta(\hat{e}) = \max[\alpha k, 1] = \begin{cases} \dfrac{e+1-\hat{e}}{e} k, & \text{kai } 1 \leqslant \hat{e} \leqslant e+1-\dfrac{e}{k}, \\ 1, & \text{kai } e+1-\dfrac{e}{k} < \hat{e} \leqslant e, \end{cases}$$

$$\text{kai } 0.99e+1 < \hat{e} \leqslant e, \quad \eta(\hat{e}) = \max[\alpha k, 1] = \begin{cases} 0.01k, & \text{kai } k \geqslant 100, \\ 1, & \text{kai } 0 < k < 100. \end{cases}$$

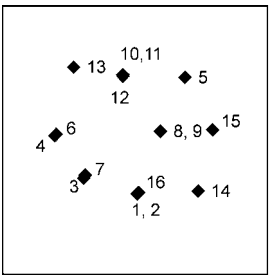
Tarp sąlygų $1 \leqslant \hat{e} \leqslant 0.99e+1$ ir $1 \leqslant \hat{e} \leqslant e + 1 - \frac{e}{k}$, kai $k < 100$, griežtesnė yra antroji, tada

$$\eta(\hat{e}) = \max[\alpha k, 1] = \begin{cases} \dfrac{e+1-\hat{e}}{e} k, & \text{kai } 1 \leqslant \hat{e} \leqslant e+1-\dfrac{e}{k}, \\ 1, & \text{kai } e+1-\dfrac{e}{k} < \hat{e} \leqslant e. \end{cases}$$

Reikia rasti slenksčio tašką $\bar{e} \in R$, kuriame $\eta(\bar{e}) = k - (n' - 1)$. Tai bus slenksčio taškas, iki kurio perskaičiuojamų neuronų kaimynystės eilė $\eta_{ij}^c = k - (n' - 1)$. Pirmoje epochoje $\hat{e}$ didesniu numeriu už $\bar{e}(\hat{e} = [\bar{e}]+1)$ kaimynystės eilė sumažės vienetu, lyginant su ($\hat{e} - 1$)-ma. Iš čia $\frac{e+1-\bar{e}}{e} k = k - (n' - 1) \Rightarrow \bar{e} = \frac{k+(n'-1)e}{k} = \frac{(n'-1)e}{k} + 1$.

Iš įrodytos teoremos seka, kad perskaičiuojamų neuronų skaičiaus priklausomybė nuo epochos numerio turi laiptinę formą. Po kiekvieno $e' = \left[\frac{n'e}{k}\right] - \left[\frac{(n'-1)e}{k}\right]$ ($n' = 1, \ldots, k-2$) skaičiaus epochų perskaičiuojamų neuronų skaičius mažėja.

SOM ir Sammono algoritmų junginys *SOMSammon*. Po SOM apmokymo gautus neuronus-nugalėtojus galima analizuoti Sammono algoritmu [2]. Sammono projekcija [7] yra netiesinis daugelio kintamųjų objektų atvaizdavimo žemesnio matavimo erdvėje



1 pav. Daugiamačių duomenų vizualizavimo pavyzdys.

metodas. Darbuose [3, 4] pateiktas SOM ir Sammono algoritmų jungimo būdas, kai daugiamačiai duomenys analizuojami Sammono algoritmu atsižvelgiant į SOM mokymosi eigą. SOM mokymo procesas, kurį sudaro e epochų, suskaidomas į pasirinktą skaičių γ blokų; po kiekvieno q-tojo ($q = 1 \ldots \gamma$) bloko gauti neuronai-nugalėtojai analizuojami Sammono algoritmu; jame pradinės dvimačių vektorių koordinatės imamos atsižvelgiant į prieš tai gautas dvimates koordinates. 1 pav. pateiktas 16-os 16-mačių vektorių vizualizavimo naudojant šį algoritmą pavyzdys (smulkiau žr. [4]). Nustatyta, kad, kuo blokų skaičius γ didesnis, tuo gaunama tikslesnė projekcijos paklaida, tačiau tokie skaičiavimai reikalauja nemažai laiko. Jiems pagreitinti tikslinga naudoti lygiagrečius algoritmus.

3. *SOMSammon* lygiagretusis algoritmas

Sudarydami lygiagretųjį algoritmą turime išskirti nepriklausomas užduotis, kurias skirtingi procesoriai gali spręsti vienu metu. Naudosime du vienodų parametrų procesorius: pirmasis procesorius skirtas SOM algoritmui vykdyti (SOM mokymui) bei duomenims kitam procesoriui paruošti; antrasis – Sammono algoritmui vykdyti. Lygiagretusis algoritmas yra realizuotas MPI funkcijų pagalba [1, 6].

Lygiagrečiojo algoritmo veikimo schema. Pirmasis procesorius atlikęs pirmąjį SOM mokymo bloką, gautus neuronus-nugalėtojus siunčia antrajam procesoriui. Antrasis procesorius, gavęs šiuos duomenis, pradeda juos analizuoti Sammono algoritmu. Tuo metu pirmasis procesorius vykdo antrąjį SOM mokymo bloką. Jį baigęs, paima iš antrojo procesoriaus dvimačius vektorius, paruošia pradines dvimačių vektorių koordinates kitam skaičiavimų Sammono algoritmu etapui. Kai tik duomenys yra paruošti, jie siunčiami antrajam procesoriui. Pirmasis procesorius tęsia SOM mokymą, antrasis – vykdo Sammono algoritmą. Taip procesas tęsiamas toliau. Kai pirmasis procesorius baigia vykdyti paskutinį SOM mokymo bloką, nusiunčia gautus neuronus-nugalėtojus antrajam procesoriui ir baigia darbą. Po to antrasis procesorius paskutinį kartą įvykdo Sammono algoritmą.

Jeigu SOM atskiro mokymo bloko ir Sammono algoritmo skaičiavimo trukmės nepriklausytų nuo bloko numerio q ir būtų vienodos tarpusavyje, tai nekiltų didelių lygiagretinimo problemų. Tačiau atskiro SOM mokymo bloko skaičiavimo trukmė mažėja augant q, tuo tarpu Sammono algoritmo skaičiavimo trukmė didėja, kadangi didėja neuronų-nugalėtojų skaičius. Problema – pasirinkus norimą SOM mokymo epochų skaičių

e ir blokų skaičių γ, parinkti Sammono algoritmo iteracijų skaičių *Sammon_iter* tokį, kad SOM vieno mokymo bloko skaičiavimo trukmė kuo mažiau skirtųsi nuo Sammono algoritmo skaičiavimo trukmės. Tada užduotys būtų kiek galima tolygiau paskirstytos tarp abiejų procesorių, o tai leistų sumažinti skaičiavimo laiką, lyginant su nuosekliuoju algoritmu. Tuo atveju bendra skaičiavimų trukmė būtų artima SOM mokymo trukmei. Siekdami išlaikyti gerą daugiamačių duomenų projekcijos kokybę, negalime per daug sumažinti Sammono algoritmo iteracijų skaičiaus: turėtume naudoti bent $Sammon\_iter \geqslant 100$.

Eksperimentiškai nustatyta, kad norint tolygiau paskirstyti užduotis tarp abiejų procesorių, t.y. suvienodinti vidutinę vieno SOM mokymo bloko ir Sammono algoritmo skaičiavimo trukmę, parametrus reikia parinkti pagal formulę $Sammon\_iter \approx 35\frac{e}{\gamma}$.

4. Tyrimo rezultatai

Kiekvieną lygiagretųjį algoritmą charakterizuoja spartinimo koeficientas S_p, įvertinantis pagreitėjimą, kurį pasiekiame spręsdami uždavinį lygiagrečiuoju algoritmu naudodami p procesorių (mūsų atveju procesorių skaičius $p = 2$), efektyvumo koeficientas E_p, parodantis, kokią dalį procesorių pajėgumo pasitelkėme spręsdami uždavinį lygiagrečiuoju algoritmu.

1 lentelėje pateiktos nuosekliojo (NA) ir lygiagrečiojo (LA) algoritmų vykdymo trukmės, spartinimo ir efektyvumo koeficientai. Matome, kad, jeigu parametrai parinkti pagal anksčiau pasiūlytą formulę, tai lygiagrečiojo algoritmo efektyvumas $E_p \geqslant 0.8$ (žr. į viršutinę lentelės dalį), priešingu atveju kompiuteriai dirba neefektyviai (žr. į apatinę lentelės dalį).

1 lentelė
Lygiagrečiojo algoritmo spartinimo ir efektyvumo koeficientai

SOM epochų skaičius e	SOM blokų skaičius γ	SOM bloko dydis e/γ	Sammono alg. iteracijų sk. *Sammon_iter*	NA, s	LA, s	S_p	E_p
300	30	10	350	67	41	1.63	0.82
300	15	20	700	66	41	1.61	0.81
300	25	12	400	66	40	1.65	0.83
300	25	12	420	67	40	1.68	0.84
400	20	20	700	86	53	1.62	0.81
400	40	10	350	88	54	1.63	0.82
400	80	5	175	91	54	1.67	0.84
300	10	30	100	37	34	1.09	0.54
300	50	6	600	128	93	1.38	0.69
400	20	20	200	57	46	1.24	0.62
400	10	40	300	54	46	1.17	0.59

5. Išvados

Pasiūlytas SOM ir Sammono algoritmų junginio lygiagretusis algoritmas, kai abu algoritmai vykdomi atskirais procesoriais. Teoriškai įrodyta, kad perskaičiuojamų neuronų skaičiaus priklausomybė nuo SOM mokymo epochos numerio turi laiptuotą formą. Be to perskaičiuojamų neuronų skaičius mažėja, didėjant vykdomos epochos numeriui, todėl vieno SOM mokymo bloko skaičiavimo laikas trumpėja. Augant mokymo bloko numeriui, neuronų-nugalėtojų skaičius didėja, todėl Sammono algoritmu tenka analizuoti vis daugiau daugiamačių vektorių, o tai pailgina skaičiavimų trukmę. Pasiūlyta vertinti vidutinę vieno mokymo bloko skaičiavimo trukmę esant fiksuotam mokymo blokų skaičiui ir parinkti tiek Sammono algoritmo iteracijų, kad vidutinė vieno SOM mokymo bloko skaičiavimo trukmė būtų beveik lygi Sammono algoritmo skaičiavimo trukmei. Eksperimentiškai nustatyta, kaip parinkti Sammono algoritmo iteracijų skaičių priklausomai nuo SOM epochų ir mokymo blokų skaičiaus. Tuo atveju lygiagretus algoritmas dirba efektyviai.

Literatūra

[1] R. Čiegis, *Lygiagretieji algoritmai*, Technika, Vilnius (2001).
[2] G. Dzemyda, Visualization of a set of parameters characterized by their correlation matrix, *Computational Statistics and Data Analysis*, **36**(1), 15–30 (2001).
[3] G. Dzemyda, O. Kurasova, Daugiamačių duomenų vizualizavimas įvertinant savireguliuojančių neuroninių tinklų mokymo eigą, *Liet. Matem. Rink.*, **42** (spec. nr.) (2002).
[4] G. Dzemyda, O. Kurasova, Visualization of multidimensional data taking into account the learning flow of the self organizing neural network, *Journal of WSCG*, **11**(1), 117–124 (2003).
`http://wscg.zcu.cz/wscg2003/Papers_2003/c11.pdf`
[5] T. Kohonen, *Self-organizing Maps*, 3rd ed., Springer Series in information Sciences, **30**, Springer–Verlag (2001).
[6] *The Message Passing Interface (MPI) Standard*.
`http://www-unix.mcs.anl.gov/mpi/`
[7] J.W. Sammon, A nonlinear mapping for data structure analysis, *IEEE Transactions on Computers*, **C-18**, 401–409 (1969).

Parallelization in combining the SOM and Sammon's mapping

G. Dzemyda, O. Kurasova, V. Marcinkevičius

In this paper, we propose a parallel algorithm for multidimensional data visualization combining the neural network (the self-organizing map-SOM) and Sammon's mapping. Here n-dimensional vectors are projected onto the plane by using Sammon's mapping taking into account the learning flow of the SOM. It is necessary to investigate some important factors that influence the efficiency of the parallel algorithm. The results of investigation allow us to optimize the number of the SOM training epochs, the number of the SOM training blocks, and the number of Sammon's iterations.