УДК 004.8:004.032.26 DOI: 10.30837/0135-1710.2022.178.033

А.Ю. ШАФРОНЕНКО, Є.В. БОДЯНСЬКИЙ

АДАПТИВНА КЛАСТЕРИЗАЦІЯ БАГАТОЕКСТРЕМАЛЬНИХ МАСИВІВ ДАНИХ З ВИКОРИСТАННЯМ МОДИФІКОВАНОГО АЛГОРИТМУ РИБ'ЯЧОЇ ЗГРАЇ

Розглянуто задачу кластеризації багатоекстремальних масивів даних. Для оптимізації функцій пошуку локальних екстремумів запропоновано алгоритм, що є по суті оптимізаційною функцією модифікованого алгоритму риб'ячої зграї, випадкового пошуку та еволюційної оптимізації. Цей алгоритм пришвидшує пошук глобальних екстремумів, не потребує додаткових обчислень, дозволяє скоротити кількість запусків процедури оптимізації, знаходити екстремуми функцій складної форми у випадку, коли класи перетинаються, та є простим у числовій реалізації.

1. Вступ

Задача кластеризації масивів спостережень довільної природи є невід'ємною частиною Data Mining, а у більш загальному випадку - Data Science. З обчислювальної точки зору, найпростішими є так звані ієрархічні методи та алгоритми, засновані на розбиттях [3], серед яких слід відзначити процедуру *k*-середніх, що набула дуже широкого розповсюдження для вирішення найрізноманітніших задач. Слід відзначити, що найбільш адекватним математичним апаратом для вирішення задач кластеризації є методи обчислювального інтелекту [5-7] і, перш за все, штучні нейронні мережі, нечіткі системи, еволюційна оптимізація та так звані гібридні системи обчислювального інтелекту, що об'єднують ці три напрями.

В загальному випадку вирішення задачі кластеризації суттєво ускладнене, якщо вихідні вектори-спостереження викривлені збуреннями, містять пропуски, самі вихідні масиви або занадто великі, або занадто короткі, кластери можуть мати досить складну форму, а їх кількість апріорі невідома. У цьому випадку найефективнішими є алгоритми, що базуються на аналізі щільностей розподілу даних [9-12], які були запропоновані для вирішення задач кластеризації великих масивів векторних даних високої розмірності. В основі цих алгоритмів лежить пошук екстремумів-максимумів функції щільності розподілу даних в масиві, що аналізується, при цьому ця функція формується як суперпозиція ядерних функцій, пов'язаних з кожним спостереженням. Фактично ця функція будується на основі вікон Парзена [13] та оцінок Надарая-Ватсона [14,15].

З обчислювальної точки зору, задача кластеризації перетворюється на проблему пошуку локальних екстремумів багатоекстремальної функції векторного аргументу щільності за допомогою градієнтних процедур, які багаторазово запускаються з різних точок вихідного масиву даних. Зрозуміло, що це займає досить багато часу, оскільки апріорі невідомо, скільки ж екстремумів має сформована функція щільності.

Пришвидшити процес пошуку цих екстремумів можна, скориставшись ідеями еволюційної оптимізації, що включає в себе алгоритми, інспіровані природою, ройові алгоритми, популяційні алгоритми тощо [16-18]. При цьому пошук ведеться одночасно групою агентів, які діють або незалежно, або у взаємодії, що дозволяє суттєво пришвидшити процес пошуку екстремумів, кожен з яких "відповідає" тому або іншому кластеру, що формується.

2. Формування функції щільності розподілу даних у масиві, що підлягає кластеризації

Вихідною інформацією для вирішення задачі кластеризації традиційно є масив векторів-спостережень $X = \{x(1), x(2), ..., x(k), ..., x(N)\}$, $x(k) = \{x_i(k)\} \in R^n$, при цьому дані попередньо відцентровано на гіперкуб так, що $x(k) = \{x_{i,i_2}(k)\} \in R^{n_1 \times n_2}$. Така ситуація може виникати у випадку обробки масивів зображень. За таких умов одними з найбільш ефективних є алгоритми, що базуються на аналізі щільностей розподілу даних у вихідних масивах, серед яких можна відзначити DENCLUE та його модифікації.

DENCLUE (DENsity-based CLUstEring) розглядається як окремий випадок оцінки ядерної щільності - це техніка непараметричного оцінювання, спрямована на пошук точок щільних областей. Основними поняттями, на яких базується DENCLUE, є функція впливу, функція щільності та атрактори щільності, що по суті є локальними екстремумами функції щільності. В загальному випадку функція впливу для будь-якого векторного спостереження $x(\bullet)$ з вихідного масиву X є ядерною дзвонуватою функцією $f^{x(\bullet)}(x)$, при цьому найпопулярнішою є традиційна гаусівська функція

$$f_G^{x(\bullet)}(x) = \exp\left(-\frac{d^2(x, x(\bullet))}{2\sigma^2}\right) = \exp\left(-\frac{\|x - x(\bullet)\|^2}{2\sigma^2}\right), \quad (1)$$

(тут $d^2(x, x(\bullet))$ - евклідова відстань; σ^2 - параметр ширини функції впливу) завдяки простоті обчислення її похідних.

У матричному випадку замість евклідової відстані можна використати метрику Флобеніуса, при цьому функція впливу набуває вигляду

$$f_G^{x(\bullet)}(x) = \exp\left(-\frac{d^2(x, x(\bullet))}{2\sigma^2}\right) = \exp\left(-\frac{Tr\left(x - x(\bullet)\right)\left(x - x(\bullet)\right)^T}{2\sigma^2}\right), \quad (2)$$

де $Tr(\bullet)$ - символ сліду матриці.

На основі функцій впливу формується функція щільності розподілу даних у масиві X у вигляді

$$f^x(x) = \sum_{k=1}^{N} f\left(x, x(k)\right). \quad (3)$$

Власне процес формування кластерів пов'язаний з відшуканням усіх екстремумів функції щільності (3) за допомогою градієнтної процедури

$$x^l = x^{l-1} + \eta^l \frac{\nabla f^x\left(x^l, x^{l-1}\right)}{\left\|\nabla f^x\left(x^l, x^{l-1}\right)\right\|}, \; x_0 = x(k), l = 0,1,2,...; \forall k = 1,2,...,N, \quad (4)$$

тобто кількість запусків алгоритму (4) визначається обсягом навчальної вибірки N. Зрозуміло, що при великих N процес кластеризації - пошуку локальних екстремумів - може потребувати дуже багато часу. Тому запропоновані модифікації DENCLUE пов'язані з пришвидшенням процесу пошуку локальних екстремумів (3) шляхом модифікації градієнтної процедури (4) [10-12].

Пришвидшити процес відшукання локальних екстремумів можна, використовуючи замість градієнтного пошуку методи еволюційної оптимізації, серед яких як достатньо ефективний, чисельно простий і швидкий можна відзначити так званий пошук на основі косяків риб [19-21], що повинен бути модифікований для вирішення задачі кластеризації.

3. Модифікований метод оптимізації на основі косяків риб

При використанні методів еволюційної оптимізації, що по суті є методами оптимізації нульового порядку, припускається, що при відшуканні екстремумів деякої функції $f^x(x)$ застосовується популяція агентів, кожен з яких діє або самостійно, або взаємодіючи з іншими, при цьому рух кожного q-го агента $(q = 1,2,...,Q)$ на l-й ітерації пошуку може бути записаний за допомогою співвідношення

$$x_q^l = x_q^{l-1} + \eta_q^l Dir_q^l, \; q = 1,2,...,Q,$$

де $x_q^l = \left(x_{q1}^l, x_{q2}^l, ..., x_{qn}^l\right)^T$, Dir_q^l - вектор, що задає напрямок руху q-го агента на l-й ітерації пошуку.

У великій родині таких методів слід відзначити метод на основі косяків риб, де кожен агент популяції імітує рух окремої риби [19-21]. Основною перевагою цього методу є

достатня ефективність відшукання глобального екстремуму досить складних функцій, до яких можна віднести і функцію щільності розподілу даних в задачах кластеризації.

Автори методу вводять у розгляд ітерації, пов'язані з рухом косяка: годування та плавання.

Оператор годування відповідає за вагу кожної риби як елемента косяка - агента. Чим важче риба, тим ближче вона до екстремума-максимума. Вага кожної риби w_q налаштовується згідно із виразом

$$w_q^l = w_q^{l-1} + \frac{f^x\left(x_q^l\right) - f^x\left(x_q^{l-1}\right)}{\max\limits_p\left\{f^x\left(x_q^l\right) - f^x\left(x_q^{l-1}\right)\right\}} \quad \forall q = 1,2,...,Q, \tag{5}$$

при цьому

$$0 < w_q^l < w_{\max}, \;\; w_l^0 = 0,5 w_{\max}.$$

Оператор плавання описує як індивідуальний рух кожної риби, так і колективний рух косяка в цілому. Тут розглядаються три типи руху: індивідуальний, інстинктивно-колективний та колективно-рольовий. Індивідуальний рух описується співвідношенням

$$x_{qi}^l = \begin{cases} x_{qi}^l + \eta_q^l Rand\{0,1\}, \text{ if } f^x\left(x_q^l\right) > f^x\left(x_q^{l-1}\right), \\ x_q^{l-1} \quad \text{else}, \end{cases} \tag{6}$$

де $Rand\{0,1\}$ - рівномірно розподілене у інтервалі $(0,1)$ випадкове число. Фактично це процедура "зондування" функції $f^x(x)$ в околі точки x_q^{l-1}, при цьому крім (6) тут може бути застосований будь-який інший алгоритм випадкового пошуку.

На базі зондування функції щільності за допомогою індивідуального руху (6) реалізується інстинктивно-колективний рух у напряму зростання цієї функції

$$x_q^l = x_q^{l-1} + \frac{\left(\sum\limits_{p=1}^{Q}\left(x_p^l - x_p^{l-1}\right)\right)\left(f^x\left(x_q^l\right) - f^x\left(x_q^{l-1}\right)\right)}{\sum\limits_{p=1}^{Q}\left(f^x\left(x_p^l\right) - f^x\left(x_p^{l-1}\right)\right)}. \tag{7}$$

Вводячи у розгляд зважений центр ваги косяка риб

$$Bar^l = \frac{\sum\limits_{p=1}^{Q} x_p^l w_p^l}{\sum\limits_{p=1}^{Q} w_p^l}, \tag{8}$$

можна записати цей рух у вигляді

$$x_q^l = \begin{cases} x_q^l - \eta_q^l Rand\{0,1\}\dfrac{x_q^{l-1} - Bar^{l-1}}{\left\|x_q^{l-1} - Bar^{l-1}\right\|}, \text{ if } \sum\limits_{p=1}^{Q} w_p^l > \sum\limits_{p=1}^{Q} w_p^{l-1}, \\ x_q^l + \eta_q^l Rand\{0,1\}\dfrac{x_q^{l-1} - Bar^{l-1}}{\left\|x_q^{l-1} - Bar^{l-1}\right\|}, \text{ if } \sum\limits_{p=1}^{Q} w_p^l < \sum\limits_{p=1}^{Q} w_p^{l-1}. \end{cases} \tag{9}$$

Для підвищення ефективності FSS у розгляд може бути введений додатковий оператор розведення, що дозволяє створювати нових риб-агентів, які мають покращені характеристики у порівнянні зі вже існуючими членами косяка. Для цього можна скористатися ідеями еволюційних операцій [23], серед яких з обчислювальної точки зору та з точки зору ефективності - надійності відшукання екстремуму можна відзначити послідовний симплекс-метод [24] та його модифікації [25-26].

Сформуємо косяк, що містить $Q = n+1$ риб-агентів, при цьому ця кількість залишається незмінною у процесі пошуку, тобто популяція $x_1^0, x_2^0, ..., x_Q^0$ генерується випадковим чином. В цій популяції знайдемо "найгіршу" рибу x_{qworst}^0, що має найменшу вагу $w_{q\min}^0$, та "найкращу" рибу x_{qbest}^0 з найбільшою вагою $w_{q\max}^0$. Основна операція руху симплекса полягає у відображенні x_{qworst}^0 через центр ваги n риб (без найгіршої), який може бути записаний у вигляді

$$\overline{x}^0 = \frac{1}{n}\sum_{q=1}^{Q}\left(x_q^0 - x_{qworst}^0\right).$$

В результаті цієї операції створюється нова риба

$$x_q^{1*} = \overline{x}^0 + \alpha\left(\overline{x}^0 - x_{qworst}^0\right),$$

яка заміняє у косяку найгіршу особину x_{qworst}^0. Так формується нова популяція $x_1^1, x_2^1, ..., x_Q^1$.

Таким чином, рух косяка-симплекса може бути описаний за допомогою співвідношень

$$\begin{cases} \overline{x}^{l-1} = \dfrac{1}{n}\sum\limits_{q=1}^{Q}\left(x_q^{l-1} - x_{qwost}^{l-1}\right), \\ x_q^l = \overline{x}^{l-1} + \alpha\left(\overline{x}_q^{l-1} - x_{qwost}^{l-1}\right), \end{cases} \tag{10}$$

що у загальному випадку є за своєю суттю алгоритмом оптимізації Нелдера-Ліда [25]. Таким чином, з косяка у процесі пошуку екстремуму вилучаються найгірші риби з найнижчою вагою та створюються нові агенти з більшою вагою.

Оскільки задача, що розглядається, є за своєю суттю проблемою багатоекстремальної оптимізації, необхідно відшукати множину екстремумів, кожен з яких є центроїдом деякого кластера. При знаходженні якогось з екстремумів з вихідної вибірки X виключаються спостереження, що розташовані безпосередньо в його околі. Після цього вилучення запропонована процедура комбінованої еволюційної оптимізації повторюється до відшукання всіх екстремумів - центроїдів.

4. Висновки

Розглянуто задачу кластеризації багатоекстремальних масивів даних. Для оптимізації функцій пошуку локальних екстремумів запропоновано алгоритм, що є по суті оптимізаційною функцією модифікованого алгоритму риб'ячої зграї, випадкового пошуку та еволюційної оптимізації. Цей алгоритм пришвидшує пошук глобальних екстремумів, не потребує додаткових обчислень, дозволяє скоротити кількість запусків процедури оптимізації, знаходити екстремуми функцій складної форми у випадку коли класи перетинаються, та є простим у числовій реалізації. Запропонований підхід дозволяє скоротити кількість запусків процедури оптимізації, дозволяє знаходити екстремуми функцій складної форми та є простим в чисельній реалізації.

Список літератури: 1. *Gan G., Ma Ch., Wu J.* Data Clustering: Theory, Algorithms and Applications. Philadiphia, Pensilvania: SIAM: 2007. 455 p. 2. *Abonyi J., Feil D.* Cluster Analisis for Data Mining and System Identification. Basel: Birlhause. 2007. 303 p. 3. *Xu R., Wunsch D.C.* II - Clustering. - Hoboken, N.J.: John Wiley Sons, Inc., 2009. 341p. 4. *Aggarwal, C.C.* Data Mining: Text Book. Springer. 2015. 5. *Engelbrecht A.P.* Computational Intelligence an Introducion. John Willey& Sons, 2007. 597 p. 6. *Rukowski L.* Computational Intelligence Methods and Techniques. Berlin Heidelberg: Springer - Verlag. 2008. 514 p. 7. *Kroll A.* Computational Intelligence. Eine Einfurming in Problelme, Methoden and Tchnische Anwen-dungen. Munchen: Oldenbourg Verlag. 2013. 428.p. 8. *Kohonen T.* Self-Organizing Maps/ Kohonen T. Berlin: Springer, 1995. 362 p. DOI: 10.1007/978-3-642-56927-2. 9. *Hinneburg A., Klim D.A.* An efficient approach to clustering in large multimedia databases with noise. Proc. 4th Int. Conf. in Knowkedge Discovery and Data Mining (KDD 98). N.Y.: AAAI Press. 1998. P. 58-65. 10. *Hinneburg A., Gabriel H.-H.* DENCLUE 2.0: Fast clustering based on kernel density estimation. 11. *Hinneburg A., Klim D.A.* A general approach to clustering in large databases with noise - kniwledge and Identification Systems. 2003. 5 (5). P. 387-415. 12. *Rehhioni H.,Idrissi A., Abourezq M., Zegrary F.* DENCLUE-IM: A new approachfor big data clustering. Procedia Computer Science. 2016. 83. P. 560-567. 13. *Parzen E.* On estimation of a proobably density function and mode. The Annalis of Math Statistics. 1962. 33. №3. P. 1065-1076. 14. *Nadaraya E.A.* On nonparametric estimation of density function and regressiion curves. Theory of Probab. Appl. 1965. 10. P. 186-190. 15.

Wantson G.S. Smooth regression analysis. Sankhya: The Indian Journal of Statistics. 1964. Ser. A. 26. № 4. P. 359-372. 16. *Kennedy J., Eberhart R.* Particle swarm optimization. Proc. IEEE Int. Conf. on Neural Networks. Perth, Australia, 1995. P. 1942-1948. 17. *Eiben A., Smith J.* Introduction to Evolutionary Computing. Heidelberg: Springer. 2003. 18. *Karpenko A. P.* Population algorithms for global continious optimization. Review of new and little - known algorithms. Приложение к журналу" Информационные технологии". 2012. № 7. 32 p. 19. *Bastos-Felino C.J.A., Lima Neto C.J.A., Lins A.J.C.C., Nascimento A.I.S., Lima M.P.* Fish School Search. Nature. Insperiod Algorithms for Optimization. Berlin Hedelberg: Springer Verlag. 2009. SCI 193 . P. 261-277. 20. *Cavalcanti Jr. G.M., Bastos-Felino C.J.A., Lima Neto F.B., Castro R.M.C.S.* A hybrid algorithm based on fish school search and particle swarm optimization for dynamic problems. Proc. Int. Conf. in Swarm Intelligence (ICSI). 2011. V. 2. P.543-552. 21. *Janecek A., Tan Y.* Feeding the fish-weight update strategies for the fish school seach algorithm. Berlin Heidelberg: Springer - Verlag. Lecture Nodes in Computer Scince. 2011. V. 6729. Part II. P. 553-562. 22. *Растрігін Л.А.* Випадковий пошук у процесах адаптації. Рига: Зінатне. 1973. 132 с. 23. *Box Y.E.P.* Evolutionary operation: A method for increasity industrial productivity. Applied Statistics. 1957. 6. P. 81-101. 24. *Spendley W., Hext G.R., Hinswath F.R.* Sequential application of simplex design in optimization and evolutionary operation. Tehnometrics. 1962. 4. P. 441-461. 25. *Nelder J.A., Mead R.* A simplex method for function minimization. Computer J. 1965. 7. P. 308-313.

Надійшла до редколегії 08.12.2022

Шафроненко Аліна Юріївна, кандидат технічних наук, доцент, доцент кафедри інформатики, ХНУРЕ. Наукові інтереси: нейронні мережі, нечітка кластеризація, еволюційні алгоритми, Data Mining, Big Data. Адреса: Україна, м. Харків, пр. Науки 14, тел. +38(068)8922587

Бодянський Євгеній Володимирович, доктор технічних наук, професор, професор кафедри штучного інтелекту, науковий керівник Проблемної НДЛ АСУ, ХНУРЕ. Наукові інтереси: обчислювальний інтелект, Data Mining, Big Data, Data Stream Mining. Адреса: Україна, м. Харків, пр. Науки 14.

УДК 004.65; 004.91 DOI: 10.30837/0135-1710.2022.178.037

Н.В. ВАСИЛЬЦОВА. І.Ю. ПАНФЬОРОВА

ДОСЛІДЖЕННЯ ВИКОРИСТАННЯ МЕТОДІВ ІЄРАРХІЧНОЇ КЛАСТЕРИЗАЦІЇ ПІД ЧАС ВИРІШЕННЯ ЗАДАЧІ АНАЛІЗУ КОНФІГУРАЦІЇ ІТ-ПРОДУКТУ

Розглянуто основні особливості існуючих способів рішення задачі аналізу конфігурації ІТ-продукту в рамках процесу управління конфігурацією. Виділено основні недоліки цих способів. Розглянуто рішення задачі аналізу конфігурації ІТ-продукту із застосуванням дивізимного та агломеративного алгоритмів. Проведено порівняльний аналіз особливостей застосування ієрархічних алгоритмів кластеризації для вирішення задачі аналізу конфігурації ІТ-продукту. Запропоновано модифікацію алгоритму найближчого сусіда, що дозволяє своєчасно виявляти конфігураційні елементи з описами, які повністю збігаються.

1. Вступ

Сучасна точка зору на процеси управління проєктами виділяє процес управління конфігурацією ІТ-продукту як один з процесів інтегрованого контролю змін. Процес управління конфігурацією спрямований на визначення конфігурації товару в окремі моменти часу. Метою даного процесу є систематичний контроль за змінами конфігурації, а також підтримка цілісності та відстеження конфігурації протягом усього життєвого циклу продукту [1]. Основними роботами процесу управління конфігурацією є: планування та управління процесом управління конфігурацією; ідентифікація конфігурації; контроль конфігурації; облік стану конфігурації; аудит конфігурації та управління випуском та доставкою продукту. Особливе місце серед цих робіт посідає робота "ідентифікація конфігурації". Дана робота визначає елементи, що підлягають контролю, встановлює схеми ідентифікації елементів та їх версій, а також інструменти та методи, які будуть використовуватися для отримання та управління виділеними елементами [1].

Проте слід визнати, що у теперішній час виділення процесу управління конфігурацією та специфікації його окремих робіт досить умовно. Рекомендації щодо впровадження процесу управління конфігурацією у життєвий цикл ІТ-продукту носять переважно теоретичний та