УДК 563.2

Ю.В. Стех, А.Б. Керницький
Національний університет "Львівська політехніка",
кафедра САПР

АЛГОРИТМИ ІЄРАРХІЧНОЇ КЛАСТЕРИЗАЦІЇ ДЛЯ ВЕЛИКИХ КОЛЕКЦІЙ ДОКУМЕНТІВ

Робота зосереджена на алгоритмах кластеризації великих колекцій документів, які базуються на ієрархічних рішеннях. Оцінена робота різних критеріальних функцій під час кластеризації документів.

This paper focuses on document clustering algorithms that build hierarchical solutions. In this paper is evaluate the performance of different criterion functions for the problem of clustering documents.

Вступ. Швидкі і продуктивні алгоритми кластеризації документів відіграють важливу роль у забезпеченні інтуїтивної навігації у колекціях документів і механізмах їх перегляду [1–3]. Це забезпечується перетворенням великих масивів інформації у ряд кластерів значно меншого розміру. Зокрема алгоритми кластеризації, які формують ієрархічні структури на основі великих колекцій документів, є ідеальними засобами для інтерактивної візуалізації і дослідження, оскільки забезпечують зображення несуперечливих і передбачуваних даних на різних рівнях деталізації [4–8]. Робота зосереджена на дослідженні алгоритмів кластеризації документів, які формують ієрархічні рішення. У статті оцінюється робота різних критеріальних функцій для проблем кластеризації документів.

Кластеризація документів. Алгоритми кластеризації використовують векторно-просторову модель для зображення кожного документа. У такій моделі, кожний документ *d* розглядається як вектор у просторі термів (інформаційних складових документа). У своїй найпростішій формі кожний документ зображується у вигляді вектора частоти входження термів у цей документ:

$$d_{tf} = (tf_1, tf_2, ..., tf_m) \tag{1}$$

де tf_i – частота входження у документ *i-го* терма. Застосовують вдосконалену схему цієї моделі. Вона полягає у зважуванні кожного терма за допомогою його оберненої частоти входження у документи у загальній колекції документів [9–14]. Причиною такого зважування є те, що терми, які часто зустрічаються у різних документах, володіють обмеженим ступенем виокремлення. Тому такі терми необхідно певним чином позбавити від спотворень. Це, як правило, здійснюється множенням частоти входження кожного терма *i* на $\log\left(N/df_i\right)$, де N – загальна кількість документів у колекції, а df_i – кількість документів, які містять і-й терм (тобто, частота документа). Це призводить до *tf-idf* зображення документа, тобто

$$d_{tfidf} = \begin{pmatrix} tf_1 \log\left(N/df_1\right), tf_2 \log\left(N/df_2\right), \dots \\ , tf_m \log\left(N/df_m\right) \end{pmatrix} \tag{2}$$

Для роботи з документами різної довжини довжина кожного вектора документа нормалізується її приведенням до одиничної довжини $\left(\left\|d_{tfidf}\right\| = 1\right)$. Це означає, що кожний документ є вектором в одинич-

ній гіперсфері. У наступній частини статті, ми припускаємо, що векторне зображення кожного документа було зважене, використовуючи *tf-idf* і є нормалізоване за допомогою приведення до одиничної довжини. Упродовж кількох останніх років було запропоновано два підходи до обчислення подібності між двома документами d_i і d_j. Перший підхід ґрунтується на застосуванні функції косинуса [4,15–17]:

$$\cos(d_i,d_j)=\frac{d_i^t d_j'}{\|d_i\|\|d_j\|} \tag{3}$$

Оскільки вектори документа виражаються через одиничні довжини, то наведена вище формула спрощується до $\cos(d_i,d_j) = d_i^t d_j$. Результати обчислень дорівнюють одиниці, якщо документи подібні, і нулю, якщо між документами немає нічого спільного (тобто, всі вектори документів є ортогональними між собою). Другий метод визначає подібність між документами, використовуючи евклідову відстань [6–8, 18]:

$$\text{dis}(d_i,d_j)=\sqrt{(d_i-d_j)(d_i-d_j)}=\|d_i-d_j\| \tag{4}$$

Якщо відстань дорівнює нулю, тоді документи є подібними, а якщо вони не подібні, тоді відстань дорівнює $\sqrt{2}$. Слід зауважити, що незважаючи на той факт, що перший підхід оцінює ступінь подібності, а другий – оцінює відстані, вони є подібними між собою, оскільки вектори документа виражаються через одиничні довжини. У статті ми використовуємо символи *n*, *m*, і *k* для позначення кількості документів, кількості умов і кількості кластерів відповідно. Використовуємо символ *S* для позначення набору з *n* документів, які ми хочемо кластеризувати, $S_1,S_2,...,Sk$ для позначення кожного з *k* кластерів, і $n_1,n_2...,n_k$ для позначення розмірів відповідних кластерів. Задану множину документів *A* і їхнє відповідне векторне зображення ми визначаємо складеним вектором D_A:

$$D_A=\sum_{d\in A} d, \tag{5}$$

тоді як центроїдний вектор C_A буде мати вигляд

$$C_A=\frac{D_A}{|A|} \tag{6}$$

Складений вектор D_A є сумою всіх векторів документів в А, а центроїдний вектор C_A – це вектор, отриманий шляхом усереднення ваг різних термів, поданих у множині документів *A*. Слід зауважити, що навіть у випадку, коли вектори документів дорівнюють одиниці, центроїдні вектори не обов'язково будуть одиничної довжини.

На вищому рівні абстракції проблема кластеризації визначається так. Задану колекцію *S*, яка складається з *n* документів, необхідно розділити на наперед визначену кількість *k* підмножин $S_1,S_2,...,S_K$ так, щоб документи, які призначені в одну підмножину, були між собою подібнішими, ніж документи, призначені у різні підмножини.

Алгоритм сегментної кластеризації використовує підхід повторної кластеризації поділом навпіл. У такому підході всі документи спочатку розподіляються у два кластери. Потім береться один із цих кластерів, що містить більше ніж один документ, і розділяється навпіл. Цей процес повторюється *n*–1 разів, призводячи до *n* листових кластерів, кожний з яких містить один документ. Легко побачити, що запропонований підхід будує агломераційне ієрархічне дерево. Внизу кожний документ знаходиться у своєму власному кластері. Ключовою характеристикою більшості алгоритмів сегментної кластеризації є те, що вони використовують глобальну оцінкову функцію, оптимізація якої впливає на весь процес кластеризації. Для таких алгоритмів сегментної кластеризації проблема кластеризації може бути визначена так: проведення процесу кластеризації так, щоб значення конкретної оцінкової функції було оптимізованим. У роботі ми досліджуємо різні оцінкові функції кластеризації. Оцінкова функція L_1 (рівняння 7) максимізує суму середніх

попарних подібностей між документами. Документи, які віднесені до кожного кластера, зважуються відповідно до розміру цього кластера.

$$L_1 = \text{maximize} \sum_{r=1}^{k} n_r \left(\frac{1}{n_r^2} \sum_{d_i, d_j \in S_r} \cos(d_i, d_j) \right) = \sum_{r=1}^{k} \frac{\|D_r\|}{n_r}. \quad (7)$$

Оцінкова функція L_2 (рівняння 8) використовує векторно-просторовий варіант алгоритму *K*-внутрішніх групових середніх. У цьому алгоритмі кожен кластер зображений його центроїдним вектором і метою є знаходження рішення, яке максимізує подібність між кожним документом і центром ваги кластера, до якого його було призначено.

$$L_2 = \text{maximize} \sum_{r=1}^{k} \sum_{d_i \in S_k} \cos(d_i, C_r) = \sum_{r=1}^{k} \|D_r\|. \quad (8)$$

Порівнюючи L_1 і L_2, бачимо, що істотною різницею між ними є те, що функція L_2 оцінює внутрішньо-кластерну подібність $\|D_r\|$. $\|D_r\|$ є квадратним коренем попарних подібностей між всіма документами у S_r і прагне передспотворити кластери, чиї документи мають меншу попарну подібність порівняно з кластерами з вищою попарною подібністю.

Оцінкова функція (рівняння 9) L_3 обчислює кластеризацію за допомогою знаходження рішення, яке відокремлює документи кожного кластера від загальної колекції. Зокрема вона намагається мінімізувати косинус між центроїдним вектором кожного кластера і центроїдним вектором загальної колекції. Внесок кожного кластера зважується пропорційно до його розміру так, що більші кластери оцінюються вище в загальному рішенні кластеризації.

$$L_3 = \text{minimize} \sum_{r=1}^{k} n_r \cos(C_r, C) = \sum_{r=1}^{k} n_r \frac{D_r^t D}{\|D_r\|}. \quad (9)$$

Запропонований алгоритм сегментної кластеризації, який використовує підхід, на якому побудований алгоритм *K*-внутрішніх групових середніх [19], оптимізує кожну із вище згаданих оцінкових функцій. На першому кроці із колекції документів вибирається випадкова пара документів, які стають центрами двох кластерів. На другому кроці для кожного документа обчислюється його подібність до цих двох центрів. Залежно від отриманих значень документи перерозподіляють у найподібніші для них кластери. Це формує початкову двосторонню кластеризацію. Надалі проводиться повторна кластеризація з метою оптимізації бажаної оцінкової функції кластеризації. Стратегія покращання результатів, яку ми використовуємо, складається із низки ітерацій. Протягом кожної ітерації, документи вибираються у довільному порядку. Для кожного документа d_i ми обчислюємо зміну в значенні оцінкової функції, одержаної шляхом переміщення d_i у інший кластер. Якщо існують такі переміщення, які приводять до покращання загального значення оцінкової функції, то d_i переміщається у кластер, який приводить до найбільшого покращання. Якщо не існує жодного переміщення, тоді d_i залишається у кластері, якому на цей момент належить. Фаза покращання закінчується, як тільки виконується ітерація, під час якої між кластерами не було переміщено жодного документа. Слід зауважити, що на відміну від традиційного підходу [19] покращання результатів, що використовується в алгоритмі *K*-внутрішніх групових середніх, запропонований алгоритм переміщає документ, як тільки він визначив, що це приведе до покращання значення оцінкової функції. Такий тип алгоритмів покращання часто називають покроковим. Оскільки кожне переміщення безпосередньо оптимізує задану оцінкову функцію, ця стратегія покращання завжди приводить до одержання локального мінімуму. Через те, що різні оцінкові функції, які використовують стратегію покращання результатів, визначаються у термінах складених кластерів і центроїдних векторів, зміна у значенні оцінкових функцій у результаті одноразового переміщення документа може бути обчислена ефективніше. Ключовим кроком у запропонованому підході повторної кластеризації шляхом поділу навпіл є метод, який визначає наступний кластер, що має бути поділений навпіл. Проводилися експерименти з двома різними

методами вибору кластерів. Перший метод використовує просту стратегію поділу навпіл найбільшого доступного у цей момент кластера. Досвід роботи з таким підходом показав, що він приводить до достатньо хороших і зважених рішень з кластеризації колекції документів. Але він має суттєвий недолік. Підхід працює не зовсім коректно на колекціях документів, в яких природні кластери є різної величини, а це зумовлює поділ спершу найбільших кластерів. Для подолання даної проблеми і отримання природніших ієрархічних рішень, ми розвинули метод, який серед поточних k кластерів вибирає кластер, який веде до $k + 1$ рішень кластеризації. Це оптимізує значення конкретної оцінкової функції (серед різних k альтернатив). Наші експерименти показали, що цей підхід працює дещо краще, ніж попередня схема. На відміну від алгоритмів сегментної кластеризації, які формують ієрархічне рішення зверху вниз, агломераційні алгоритми формують рішення, призначаючи спочатку кожен документ його власному кластеру. Після цього ітераційно проводиться вибирання і об'єднання пар кластерів, для отримання єдиного комплексного кластера. Агломераційні алгоритми формують дерево знизу (з листків) вгору (тобто до кореня). Ключовий параметр в агломераційних алгоритмах є метод, який використовується для визначення пари кластерів, які необхідно об'єднати на даному кроці. У більшості агломераційних алгоритмах це здійснюється, вибираючи найподібнішу пару кластерів. На теперішній час розроблені численні підходи для обчислення подібності між двома кластерами. У нашому дослідженні ми використовували такі схеми: єдиного зв'язку, повного зв'язку, групових середніх, а також критеріальні функції, описані вище. Схема єдиного зв'язку визначає подібність двох кластерів за максимумом подібності між документами з кожного кластера. Тобто, схожість між двома кластерами S_i і S_j визначається так:

$$\mathrm{sim}_{\text{single-link}}(S_i,S_j)= \max_{d_i\in S_i, d_j\in S_j} \left(\cos(d_i,d_j)\right) \quad . \tag{10}$$

На відміну від попереньої схеми, схема повного зв'язку використовує мінімальну подібність між парою документів:

$$\mathrm{sim}_{\text{complete-link}}(S_i,S_j) = \min_{d_i\in S_i, d_j\in S_j} \left(\cos(d_i,d_j)\right). \tag{11}$$

Загалом як схема єдиного зв'язку, так і схема повного зв'язку не працюють дуже добре, тому що вони базуються на обмеженому об'ємі інормації (у випадку схеми єдиного зв'язку), або припускають, що всі документи у кластері дуже подібні між собою (у випадку схеми повного зв'язку). Схема групових середніх вирішує ці проблеми, обчислюючи подібність мід двома кластерами як середню величину попарної подібності документів з кожного кластера:

$$\mathrm{sim}_{\text{UPGMA}}(S_i,S_j) = \frac{1}{n_i n_j} \sum_{d_i\in S_i, d_j\in S_j} \cos(d_i,d_j) = \frac{D_i^t D_j}{n_i n_j}. \tag{12}$$

Оцінкові функції, описані в алгоритмах сегментної кластеризації, можуть бути перетворені на кластерні схеми вибору для агломераційної кластеризації, використовуючи загальну схему покрокової оптимізації. Розглянемо колекцію з n-документів і рішення кластеризації, яке було обчислене після виконання l кроків об'єднання кластерів. Це рішення міститиме $n–l$ кластер, оскільки кожен об'єднувальний крок скорочує кількість кластерів на одиницю. Тепер у цьому $(n - l)$ рішенні кластеризації, вибирається пара кластерів для об'єднання, що приводить до $(n - l - 1)$ рішення, яке оптимізує визначену оцінкову функцію. Тобто, кожна з $(n - l) \times (n - l - 1)/2$ пар можливого об'єднання оцінюється, і вибирається те, яке сформує рішення кластеризації, яке має максимальне (або мінімальне) значення визначеної оцінкової функції. Оцінкова функція є локально оптимальною у межах даного кроку агломераційного алгоритму. Цей процес продовжується доки не отримаємо повне агломераційне дерево.

Висновки. Запропонований алгоритм сегментної кластеризації оптимізує оцінкові функції

$$L_1 = \text{maximize} \sum_{r=1}^{k} n_r \left(\frac{1}{n_r^2} \sum_{d_i,d_j\in S_r} \cos(d_i,d_j) \right) = \sum_{r=1}^{k} \frac{\|D_r\|}{n_r}, \quad L_2 = \text{maximize} \sum_{r=1}^{k} \sum_{d_i\in S_k} \cos(d_i,C_r) = \sum_{r=1}^{k} \|D_r\|,$$

L_3=minimize $\sum_{r=1}^{k} n_r \cos(C_r, C) = \sum_{r=1}^{k} n_r \frac{D_r' D}{\|D_r\|}$. На відміну від традиційного підходу покращання результатів, що використовується в алгоритмі K-внутрішніх групових середніх, запропонований алгоритм переміщає документ, як тільки відомо, що це приведе до покращання значення оцінкової функції. Оскільки кожне переміщення безпосередньо оптимізує конкретну оцінкову функцію, ця стратегія покращання завжди спричиняє одержання локального мінімуму. Різні оцінкові функції, які використовують стратегію покращання результатів, визначаються у термінах складених кластерів і центроїдних векторів, тому зміна у значенні оцінкових функцій внаслідок одноразового переміщення документа може бути обчислена швидше.

Розвинуто метод, який серед поточних k-кластерів, вибирає той, який веде до k + 1 рішень кластеризації. Це оптимізує значення конкретної оцінкової функції (серед різних k альтернатив). Наші експерименти показали, що цей підхід працює дещо краще, ніж попередня схема.

1. Charu C. Aggarwal, Stephen C. Gates, and Philip S. Yu. On the merits of building categorization systems by supervised clustering. In Proc. of the Fifth ACM SIGKDD Int'l Conference on Knowledge Discovery and Data Mining, – 1999. – p. 352–356. 2. Doug Beeferman and Adam Berger. Agglomerative clustering of a search engine query log. In Proc. of the Sixth ACM SIGKDD Int'l Conference on Knowledge Discovery and Data Mining, – 2000. – p. 407–416. 3. Daniel Boley. Principal direction divisive partitioning. Data Mining and Knowledge Discovery, 2(4), 1998. 4. Chung-Kuan Cheng and Yen-Chuen A. Wei. An improved two-way partitioning algorithm with stable performance. IEEE Transactions on Computer Aided Design, 10(12):1502–1511, December 1991. 5. Inderjit S. Dhillon and Dharmendra S. Modha. Concept decompositions for large sparse text data using clustering. Machine Learning, 42(1/2):143–175, 2001. 6. R.O. Duda, P.E. Hart, and D.G. Stork. Pattern Classification. John Wiley & Sons, 2001. 7. A.K. Jain and R. C. Dubes. Algorithms for Clustering Data. Prentice Hall, 1988. 8. B. King. Step-wise clustering procedures. Journal of the American Statistical Association, 69:86–101, 1967. 9. Bjornar Larsen and Chinatsu Aone.Fast and effective text mining using linear-time document clustering. In Proc. of the Fifth ACM SIGKDD Int'l Conference on Knowledge Discovery and Data Mining, – 1999. – p. 16–22. 10. R. Ng and J. Han. Efficient and effective clustering method for spatial data mining. In Proc. of the 20th VLDB Conference, pages 144–155, Santiago, Chile, 1994. 11. J. Puzicha, T. Hofmann, and J. Buhmann. A theory of proximity based clustering: Structure detection by optimization. PATREC: Pattern Recognition, Pergamon Press, 33(4):617–634, 2000. 12. G. Salton. Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer. Addison-Wesley, 1989. 13. Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(8):888–905, 2000. 14. K. Zahn. Graph-tehoretical methods for detecting and describing gestalt clusters.IEEE Transactions on Computers, (C-20):68–86, 1971. 15. Батыршин И.З., Хабибулин Р.Ф. Тестирование кластерных алгоритмов на инвариантность относительно нумерации объектов. // Известия академии наук. Теория и системы управления. – 1997, 2, 165 – 168. 16. Батыршин И.З., Хабибулин Р.Ф. Разработка алгоритмов когнитивного кластерного анализа, в кн.: Обработка текста и когнитивные технологии, вып. 3 / Под ред. Соловьева В.Д. – Пущино, 1999, с. 43 – 47. 17. Ермаков А.Е. Тематический анализ текста с выявлением сверхфразовой структуры // Информационные технологии. – 2000. – № 11. 18. Ермаков А.Е., Плешко В.В. Ассоциативная модель порождения текста в задаче классификации // Информационные технологии. – 2000. – N №. 19. Харламов А.А., Ермаков А.Е., Кузнецов Д.М. Технология обработки текстовой информации с опорой на семантическое представление на основе иерархических структур из динамических нейронных сетей, управляемых механизмом внимания // Информационные технологии. -- 1998. – № 2. – С. 26–32.