Методи факторизації тензорів при пошуку семантичної відстані між текстами, написаними природньою мовою

Смєлов Валерій Вікторович

Київський Національний Університет імені Тараса Шевченка

Факультет кібернетики Кафедра математичної інформатики

smielov.knu@gmail.com

Анотація

Для аналізу текстів, написаних природньою мовою, потрібно виконати перехід від тексту до матриці. Оскільки тексти можуть мати великі розміри, то й, відповідно, розмірність матриці може бути значною, отже, виникає потреба у наближеному представленні даних. У статті розглянуто спосіб факторизації матриць, наведено математичне обґрунтування та запропонована програмна реалізація.

1. Вступ

Оскільки швидкість виникнення інформації у текстовому вигляді є надзвичайно великою і продовжує зростати, все більш актуальною стає проблема програмного аналізу текстів, написаних природньою мовою. Сьогодні існує багато алгоритмів, які дозволяють проводити таку первинну обробку текстів як визначення мови, якою написано текст, визначення слів, які зустрічаються найчастіше, належність тексту до певної категорії відповідно до заданих параметрів. Проте, в галузі аналізу текстів, написаних природньою мовою, досі існує багато нерозв'язаних проблем. Для деяких існують алгоритми, які працюють на певних наборах вхідних даних, інші ж мають жорстокі обмеження щодо обсягу вхідних даних. Однією з таких проблем є визначення семантичної відстані між заданими текстами: на вході програма отримує два (або більше) тексти, написані однією й тією ж мовою; на виході програма повертає коефіцієнт, який вказує наскільки об'єкти, події чи явища, описані в текстах, семантично близькі один до одного.

Семантична близькість – це величина, що показує наскільки два поняття пов'язані (або схожі) між собою. Обчислення семантичної близькості має надзвичайно широке застосування у комп'ютерній лінгвістиці, наприклад: семантичний аналіз, розв'язання полісемії та анафор, кластеризація та класифікація текстів, ідентифікація сутностей в тексті та ін. Для визначення семантичної близькості між текстами слід провести аналіз обох текстів, та на підставі проведеного аналізу вказати ступінь близькості. Для проведення аналізу тексту потрібно здійснити перехід від природньої мови до формальної моделі, такою моделлю зазвичай виступає матриця. Примітивний підхід полягає у переході до двомірної термін-документ матриці, але для встановлення більш чітких зв'язків використовується тримірна матриця підмет – присудок – додаток. Такий підхід дозволяє уникнути зіставлення різних значень одного й того ж слова. Наприклад, порівняння казкового персонажу "принцеса" з особою з королівської сім'ї Великої Британії. Завдяки порівнянню слів, з якими вступає у зв'язок слово "принцеса" виникає можливість уникнення ототожнення цих понять. Але оскільки розміри матриць залежності слів є значними за розмірами то постає питання невід'ємної факторизації тензорів великих розмірів.

2. Математична частина

Проблему можна записати у такому вигляді:

Є $G \in R^{R\times S\times T}$ – тривимірний не негативний тензор, який потрібно проаналізувати. Необхідно мінімізувати наступний вираз:

$$\min_{\hat{G}>0} \left\| G - \hat{G} \right\|_F^2$$

де $\hat{G}$ – шукана матриця, а $\|\mathrm{A}\|_F^2$ – норма Фребеніуса, яку знаходимо за такою формулою:

$$\|\mathrm{A}\|_F = \sqrt{\sum_u^R \sum_v^S \sum_w^T \left| a_{u,v,W} \right|^2}$$

Матриця $\hat{G}$ має бути представленна у вигляді:

$$\hat{G} = \sum_{k=1}^{K} u^{(k)} \otimes v^{(k)} \otimes w^{(k)},$$

де $u^{(k)} \in \mathbb{R}^R$, $v^{(k)} \in \mathbb{R}^S$ та $w^{(k)} \in \mathbb{R}^R$ - базові вектори з невід'ємними значеннями. Найбільш поширений підхід до невід'ємної факторизації тензорів заснований на методі Блоків Гуесса-Сайдела (БГС). Використання комбінації методів Блоків Гуесса-Сайдела та ітерації Якобі модифікує схему таким чином:

$$u_i^{(k)} \leftarrow \frac{u_i^{(k)} \sum_{s,t} G_{i,s,t} v_s^{(k)} w_t^{(k)}}{\sum_{m=1}^{K} u_i^{(m)} \langle v^{(m)}, v^{(k)} \rangle \langle w^{(m)}, w^{(k)} \rangle}$$

$$v_i^{(k)} \leftarrow \frac{u_i^{(k)} \sum_{s,t} G_{i,s,t} u_s^{(k)} w_t^{(k)}}{\sum_{m=1}^{K} u_i^{(m)} \langle u^{(m)}, u^{(k)} \rangle \langle w^{(m)}, w^{(k)} \rangle}$$

$$w_i^{(k)} \leftarrow \frac{u_i^{(k)} \sum_{s,t} G_{i,s,t} v_s^{(k)} w_t^{(k)}}{\sum_{m=1}^{K} u_i^{(m)} \langle u^{(m)}, u^{(k)} \rangle \langle v^{(m)}, v^{(k)} \rangle},$$

де G – заданий тензор, а $\langle x, y \rangle$ – позначає скалярний добуток векторів.

Зазвичай, наведена ітеративна процедура може повторитися тисячу чи навіть сотню тисяч разів, перш ніж наблизиться до шуканого розв'язку. Це залежить від складності вхідних даних і кількості бажаних ітерації. Таким чином, ітеративний підхід до невід'ємної факторизації тензорів є досить ємким за часом, тож слід запропонувати певні оптимізації, які дозволять значно пришвидшити процес.

3. Програмна реалізація

Алгоритм, який буде описано, заснований на теорії, що була наведена раніше. Першим кроком алгоритму буде ініціалізація векторів *u, v, w* початковими значеннями, які обираються випадковим чином з проміжку 0..1. Основна задача буде полягати у перерахуванні значень *u, v, w* відповідно до зазначених в математичній частині формул. Скалярний добуток можна буде обрахувати заздалегідь, і зберегти в матриці розмірності $K \times K$. Напишемо формулу для обрахування M_u, значення елементів матриць M_v та M_w будуть визначатися аналогічним до наведеного подання чином:

$$M_u = \begin{bmatrix} \langle u^{(1)}, u^{(1)} \rangle & \cdots & \langle u^{(1)}, u^{(K)} \rangle \\ \vdots & \ddots & \vdots \\ \langle u^{(K)}, u^{(1)} \rangle & \cdots & \langle u^{(K)}, u^{(1)} \rangle \end{bmatrix}$$

Оскільки матриця M_u є симетричною, то для оптимізації можна обраховувати лише верхній чи нижній трикутник. Така оптимізація дасть змогу скороти обчислення вдвічі.

Псевдокод алгоритму матиме вигляд:

Вхід: G – матриця розмірності $R \times S \times T$, Ранг K та кількість ітерацій I

Вихід: *u, v, w*

1. Ініціалізація *u, v, w* випадковими числами
2. M_u ← Перерахунок(u)
3. M_v ← Перерахунок(v)
4. M_w ← Перерахунок(w)
5. for i ∈ {0,.., $I-1$} do
6. u← Крок(G,u,v,w, M_v, M_w)
7. M_u← Перерахунок(u)
8. v← Крок(G,u,v,w, M_u, M_w)
9. M_v ← Перерахунок(v)
10. w← Крок(G,u,v,w, M_u, M_v)
11. M_w ← Перерахунок(w)
12. end for
13. return u,v, w

де "Перерахунок" позначає операцію перерахування матриці скалярних добутків, "Крок" – операцію знаходження нових значень шуканих векторів.

Оскільки перерахування складових векторів *u, v, w* та матриць M_u, M_v та M_w є незалежним, то існує можливість використання методів паралельних розрахунків для відшукання зазначених величин.

4. Тестування

Тестування було проведено на комп'ютері з процесором Intel Core i7-920 та 6 GB DDR3 RAM. У якості тестів генерувалися випадкові набори даних. Розміри тензорів (R=S=T) варіювалися від 100 до 1000. На одному наборі даних проводилося 5 однакових тестів. У результуючу таблицю записаний середній арифметичний час проведених тестів. На кожному наборі даних проводилося 500 ітераційних кроків. Час вимірювався в мілісекундах. У таблиці час наведено в секундах (з точністю до двох знаків після десяткової коми). Результати роботи програми зображені у Таблиці 1.

Таблиця 1. Швидкодія програми

R=S=T	Час виконання (с)
100	0,23
300	1,54
500	4,07
600	6,41
700	7,83
800	10,28
900	13,75
1000	16,92

5. Висновок

Після проведення факторизації, асимптотичний час доступу до елементу має лінійну складність O(N), в той час, як затрати на пам'ять для представлення зменшуються з O(N^3) до O(N^2), що є надзвичайно важливим. Оскільки процес "відтворення" елементу матриці не має розгладжень то він може бути здійснений за допомогою методів паралельного обчислення.

У статті було наведено основні причини походження проблеми невід'ємної факторизації тензорів при обробці текстів написаних природньою мовою, математичне обґрунтування розв'язку та псевдокод програми. Також було проведено тестування коректності алгоритму.

Подальша робота може бути проведена в напрямку написання системно-залежного розв'язку, який би використовував особливості паралельної роботи конкретної системи для отримання кращих результатів.

Література

1. *Nonnegative Tensor Factorization Accelerated Using GPGPU* -Jukka Antikainen, Jırı´ Havel, Radovan Josth, Adam Herout, Pavel Zemcı´k

2. A. *Andriyashin, J. Parkkinen, and T. Jaaskelainen, "Illuminant Dependence of PCA, NMF and NTF in Spectral Color Imaging,"Proc*. 19th Int'l Conf. Pattern Recognition (ICPR '08), pp. 1-4, Dec.

3. Lee D.D. and Seung H.S. *Algorithms for Non-Negative Matrix Factorization* // NIPS, 2000.

4. Pauca V. et al. *Text Mining Using Non-Negative Matrix Factorizations*. Proc.4th SIAM Intl.Conf.Data Mining. – 2004.