Латентен семантичен анализ

Латентният семантичен анализ (latent semantic analysis, LSA), известен още и като латентно семантично индексиране (latent semantic indexing, LSI) [1] представлява метод за намаляване на размерността на векторите, описващи документите в дадена колекция, който открива взаимовръзките (подобията) не само между отделните документи, но и между думите, които ги съставят. Последното е основната му разлика със значително по-лесния за реализация векторен модел за анализ на текст (vector space model, VSM). Презумпцията е, че *думите* (*terms*), които имат подобно значение често се срещат заедно в едни и същи документи. В това има логика, защото когато човек пише текст се старае да избегне многократното дублиране на думи, в резултат на което често ги занемя със синоними или други изразни средства, имащи подобно значение. В този смисъл LSA може да идентифицира и в последствие групира семантично-свързаните думи в *по-общи „теми“ (topics)*.

Методът се нарича *латентен семантичен анализ*, тъй като въпросните теми всъщност са скрити (*латентни*), т.е. не са явно посочени, нито наименувани, но методът стига до тях чрез анализ на семантично-свързаните думи, които открива и групира. Откритите теми по същество представляват концепции, на които обаче методът не може да даде някакво конкретно заглавие на естествен език. Т.е. на алгоритмично ниво концепциите остават неозаглавени, но ако човек, който ги интерпретира реши – винаги може да им даде някакво название. *Всяка уникална дума в колекцията от документи е свързана до определена степен (малко или много, а може и никак) с някоя тема, а от своя страна всяка тема има определен принос (дял) в описанието на всеки от документите.* Ако трябва отново да се направи паралел с векторния модел за анализ на текст – при него смисълът на документите се извлича от думите, които ги съставят. При LSA, смисълът на документите се извлича от откритите латентни теми, които обаче са получени чрез групирането на семантично-свързаните думи. Т.е. *латентните теми представляват едно междинно, неявно (скрито) ниво на описание на документите*, които вече се описват не чрез думите, а чрез темите. И тъй като в рамките на всяка колекция от документи, откритите *значими* латентни теми са многократно по-малко от думите в колекцията, от там идва и намалението в размерността на векторите, описващи документите.

Например думите: космос, ускорител, совалка, ракета, сонда и планета образуват *„космическа тема“*. Даден документ е свързан с космическата тема, ако съдържа коя да е, една или повече, от тези думи. Така два или повече документа могат да се идентифицират като семантично свързани, дори и да не съдържат общи думи. За сравнение, при VSM, всяка дума се третира като независима от другите и ако в един документ се говори за ракети, а в друг за совалки или междупланетни сонди, то семантичната близост между тях ще бъде нула. Но не и при LSA. В този смисъл, латентният семантичен анализ надеждно се справя със *синонимите* и от части с *полисемията* (многозначността на думите), което е и най-голямото му предимство пред векторния модел за анализ на текст. И тъй като думите с подобно значение биват групирани в общи теми, то предварителната обработка на текста може

значително да се олекоти като се пропусне стемирането на думите, т.е. отделянето на корена от окончанията. На теория то не е необходимо, защото със или без стемиране, съответните думи ще влязат в една и съща тема, независимо дали като една или повече думи.

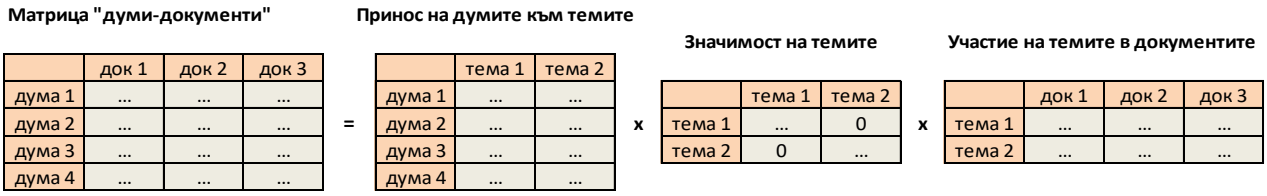
Като входни данни, LSA приема *матрицата думи-документи* (term-document matrix) – най-лявата част на фигура 1. В нея думите са разположени по редове, а документите по колони. В общия случай тя показва колко пъти се среща всяка дума във всеки един документ, но стойностите в матрицата могат да бъдат и tf-idf теглата на думите по отношение на отделните документи. Собствените ми експериментални изследвания показват, че ако в матрицата се използват именно tf-idf теглата на думите, вместо суровите честоти на поява (броя срещания на думите в документите), изчислените коефициенти на подобие между отделните документи или между заявката и документите са значително по-точни.

Нека матрицата „думи-документи“ се нарича A. Тогава редът a_i съдържа теглата на *i*-тата дума по отношение на всички документи. Аналогично, редът a_p съдържа теглата на *p*-тата дума по отношение на същите документи. Тогава скаларното произведение между двата реда (вектора) $a_i^T a_p$ ще даде скалар, посочващ доколко свързани са тези две думи *i* и *p*. А ако върху него се приложи косиносува нормализация, то ще се получи коефициентът на подобие между думите. Именно по този начин LSA определя взаимовръзките между отделните думи в колекцията от документи. По същия начин, ако се изчисли матричното произведение AA^T ще се получи матрица, съдържаща коефициентите на подобие между всички комбинации от думи в цялата колекция от документи. Аналогично, произведението $A^T A$ ще даде подобията между всички документи в колекцията.

Намирането на подобията между отделните думи е важно, но по никакъв начин не групира семантично свързаните и не води до откриване на латентните теми. Последното се извършва с помощта на математическа операция, наречена *разлагане на матрици по особени (сингулярни) стойности* (*singular value decomposition* или SVD). Според тази операция, всяка правоъгълна матрица A може да се разложи до следното матрично произведение:

$$A = U\Sigma V^T \quad (1)$$

където U и V са ортогонални, а Σ е диагонална матрица, т.е. всичките ѝ стойности извън главния диагонал са нули. Стойностите на Σ се наричат *особени* или *сингулярни стойности*. Подредени са в низходящ ред по главния диагонал и показват нивото на значимост на откритите латентни теми. Стойностите на U се наричат *леви особени (сингулярни) стойности* и показват *приноса на всяка една дума към всяка скрита тема*. Стойностите на V^T се наричат *десни сингулярни стойности* и показват *участието (приноса) на всяка открита тема във всеки един от документите* в колекцията. Идеята е представена на фиг. 5.5 с 4 думи, 3 документа (A_{4x3}) и 2 скрити теми.



Фигура 1. Разлагане (SVD) на матрицата „думи-документи“ до три матрици, показващи нивото на значимост на всяка скрита тема, както и приноса на думите към темите и участието на темите в документите.

Тук трябва да се отбележи, че ако A има размерност $m_x n$, то размерността на U е $m_x m$, размерността на Σ е $m_x n$ и размерността на V^T е $n_x n$. Но нали LSA е метод за намаляване на размерността на векторите, описващи документите, т.е. на матрицата „думи-документи“. Затова се вземат само *най-високите k на брой особени* (сингулярни) *стойности*, т.е. *най-значимите k на брой скрити теми*, и съответните им особени вектори от U и V. По този начин се реализира т.нар. *съкратено разлагане по особени стойности* (truncated SVD). Така размерността на U става $m_x k$, на Σ - $k_x k$ и на V^T - $k_x n$. Разбира се, при съкратеното SVD, след разлагането трите матрици представляват достатъчно точна апроксимация, но не и идентично представяне на оригиналната матрица.

Може да не се концентрирате особено върху математиката. Няма да има формули на изпита. За LSA може да има само по-общи (концептуални) въпроси. Алгоритмите за реализация на SVD и методите за изчисляване на детерминанти също няма да присъстват в изпита.

От линейната алгебра е известно, че колоните на U всъщност са *собствените вектори* (*eigenvectors*) на матричното произведение AA^T, колоните на V (или редовете на V^T) са *собствените вектори* на произведението $A^T A$, а особените стойности на Σ са корен квадратен от *собствените стойности* (*eigenvalues*) на AA^T или $A^T A$. С други думи, за да се реализира разлагането по особени (сингулярни) стойности, трябва да се намерят собствените стойности и собствените вектори матричните произведения AA^T и $A^T A$.

От линейната алгебра също е известно и следното уравнение:

$$Av = \lambda v \qquad (2)$$

където A е матрица, v е *собствен вектор*, а λ е *собствена стойност* на матрицата A. Уравнение (2) може да се пренапише и във вида на (3)

$$(A - \lambda E)v = 0 \qquad (3)$$

където E е единичната матрица.

Уравнение (3) може да има ненулев собствен вектор v, само ако детерминантата на матрицата $(A - \lambda E)$ е нула. Т.е.

$$|A - \lambda E| = 0 \text{ или } \det(A - \lambda E) = 0 \qquad (4)$$

Уравнение (4) се нарича характеристичен полином на матрицата A. Представлява полином на λ от n-та степен, където n е рангът на матрицата. Това означава, че той има

максимум *n* на брой различни решения за λ, т.е. *n* на брой собствени стойности. На всяка собствена стойност λ съответства ненулев собствен вектор v. Всъщност, при решаване на системата от линейни уравнения (3) чрез заместване на собствената стойност, могат да се получат множество собствени вектори, съответстващи на една и съща собствена стойност. Но всички тези вектори са в едно направление, затова за една собствена стойност се взема само един съответстващ собствен вектор и то такъв, че *собствените вектори* на матричните произведения AA^T и A^TA *да бъдат ортонормирани*.

Вече стана ясно, че за да се реализира разлагането на матрицата A по особени стойности (SVD), трябва да се изчислят собствените стойности и собствените вектори на матричните произведения AA^T и A^TA. Това става чрез намиране на детерминанта (4), което обаче е изчислително-интензивна задача, особено при детерминанти от висок ред.

Съществуват различни начини за изчисляване на детерминанти на матриците. Някои се изучават в курсовете по висша математика – например метод на адюнгираните количества (разширение на Лаплас, Laplace expansion), Гаусова елиминация (Gaussian elimination), метод на Гаус-Жордан (Gauss–Jordan elimination) и др. Методът на адюнгираните количества е подходящ за демонстрационни и учебни цели, тъй като е лесен за разбиране, но времевата му сложност от $O(n!)$ го прави изцяло неприложим за практическо смятане на детерминанти от висок ред. Гаусовата и Гаус-Жордановата елиминация имат значително по-добра времева сложност от $O(n^3)$ и техни модификации стоят в основата на някои от най-използваните алгоритми и методи за изчисление на детерминанти. Например, LU разлагането (lower–upper decomposition), предложено от Tadeusz Banachiewicz се базира на Гаусова елиминация [2]. Алгоритъмът на Erwin Bareiss [3] също е вариант на Гаусова елиминация. И двата се характеризират с времева сложност от $O(n^3)$. Най-бързият известен алгоритъм за изчисляване на детерминанта е бързото умножение на матрици (Fast matrix multiplication) на James Bunch и John Hopcroft [4,5] със сложност $O(n^{2.373})$.

Съществуват и алгоритми, които директно намират собствените стойности и собствените вектори. Например *QR алгоритъмът* [6,7,8], предложен в началото на 60-те години на миналия век от John Francis и Vera Kublanovskaya, или алгоритъмът, кръстен на Carl Gustav Jacobi [9], който предлага подобен метод още през 1846 г. За разлагане на матрици по особени стойности (SVD) обикновено се използва версията на QR алгоритъма, предложена от Gene H. Golub и William Kahan [10] през 1965 г.

След намиране на собствените стойности и собствените вектори на матричните произведения AA^T и A^TA: корен квадратен от собствените стойности, подредени по големина, образуват главния диагонал на Σ; собствените вектори на AA^T, позиционно съпоставени с подредените собствени стойности, формират колоните на U; и собствените вектори на A^TA, отново позиционно съпоставени със собствените стойности – колоните на V (или редовете на V^T). За да се реализира съкратено SVD разлагане, в матриците U, Σ и V остават само тези редове или колони, които отговарят на първите k на брой, най-високи, особени (сингулярни) стойности, т.е. на първите k на брой най-значими латентни/скрити теми. Разбира се, съкратеното SVD разлагане вече не представлява точна декомпозиция на оригиналната матрица A, а получените матрици U_k, Σ_k и V_k^T отговарят на разложената по особени стойности матрица A_k, която се явява апроксимация от ранг k на оригиналната A.

Съвсем естествено при тази апроксимация възниква някаква, макар и минимална, грешка, но пък така се постига редуциране в размерността на документните вектори и преминаване от пространство на множеството несвързани думи към по-семантично ориентираното пространство на темите.

И накрая, за да се изчисли *степента на подобие между кои да са два документа* в колекцията, на базата на *редуцирания брой скрити теми*, трябва просто да се пресметне *косинуса на ъгъла между съответните две колони* във V_k^T. Ако е необходимо да се изчисли подобие между заявка q и някой от документите в колекцията, при положение, че q не присъства във V_k^T, то първо q трябва да се трансформира във вектор с k на брой измерения q_k (отчитайки само най-значимите k на брой теми), за да притежава същата размерност като колоните на V_k^T. След това подобието отново се намира чрез косинуса на ъгъла между q_k и съответната колона във V_k^T. Трансформацията на q в q_k става чрез матричното уравнение (5).

$$q_k = \Sigma_k^{-1} U_k^T q \qquad (5)$$

От тук надолу, пак може да четете по-внимателно.

Една от големите цели на латентния семантичен анализ е да изчисли степен на подобие между два документа или между заявка и документ. Върху точността на изчисленото подобие обаче влияят редица фактори, като най-значимите от тях са:

- Броят латентни (скрити) теми.
- Начинът, по който са изчислени елементите на матрицата „думи-документи“. Дали това са просто честотите на поява на отделните думи в съответните документи или tf-idf тегла.
- Моделите за изчисляване на теглата на думите, ако стойностите в матрицата са tf-idf тегла.
- Дали думите са предварително *стемирани* или не.

Броят на използваните латентни теми влияе съществено върху точността на изчислените подобия, но за съжаление няма теоретично-мотивирано правило, с което този брой предварително да се фиксира. Той зависи силно от други фактори, като броя на документите в колекцията, броя на уникалните думи (термини) в речника, начина на формиране на теглата на думите и т.н. Най-точният начин за определяне на подходящия брой скрити теми, за дадена задача или колекция от документи, е експериментално, което до някъде е проблем. Използването на твърде голям брой теми, ще направи латентния семантичен анализ (LSA) да се държи като векторния модел за анализ на текст (VSM), третирайки думите като отделни и независими. От друга страна, използването на твърде малък брой латентни теми ще накара LSA да започне да групира семантично-несвързани думи в общи теми, което също ще доведе до загуба на точност. Според експерименталните данни на учени, за колекция от около 5000 документа и матрица, съдържаща само честотата на поява на отделните думи в съответните документи, стойност от 100 е добра отправна точка за експериментално определяне на оптималния брой скрити теми. Логично е, че ако броят на уникалните думи и документите в колекцията се увеличи/намали, оптималният

брой латентни теми също би трябвало да се увеличи/намали. Собствени експериментални изследвания изцяло потвърждават това предположение.

По принцип латентният семантичен анализ се извършва върху матрица „думи-документи“, съставена от броя появи на всяка една от думите в съответните документи. Но елементите ѝ могат да бъдат и tf-idf тегла. Дори резултатите от проведените експериментални изследвания показват, че изчислените коефициенти на подобие са с около 5-7 процентни пункта (ppts) по-високи, ако матрицата е съставена от tf-idf тегла. Като за разлика от VSM, при латентния семантичен анализ се постига по-висока точност на изчислените подобия, ако IDF се приложи и към думите в заявката, и към думите в документите.

Както вече стана ясно, стемирането на думите не би трябвало да оказва съществено влияние върху точността на изчислените подобия, тъй като по идея LSA групира семантично свързаните думи в общи теми. И дали множество производни на една дума ще бъдат обобщени в една тема, или множеството стемирани думи ще се разпознаят като една дума, е почти еднакво. Почти, но не съвсем, защото при първия случай множеството думи, макар и групирани, все пак си остават множество думи, докато при стемирането се разпознават като една единствена. И ако към тях се групират и други семантично свързани думи, тогава биха се появили малки нюанси – но наистина малки, които няма съществено да променят изчислените подобия.

По отношение на точността на изчислените коефициенти на подобие са проведени подробни експериментални изследвания и сравнение с останалите разгледани методи за описание на документи и анализ на текст. Очакването е LSA да се представя по-добре от VSM, основно поради възможностите за справяне със синоними и групирането на семантично свързаните думи. Редица автори съобщават за проведени експериментални изследвания и сравнителни анализи на LSA и VSM, приложени върху стандартизираните множества от данни (datasets) MED, CISI, NPL, TIME и CACM, които се използват за оценка на различните методи за търсене и извличане на документи. При голяма част от тях, резултатите от LSA и VSM са съпоставими. Но при колекцията MED, латентният семантичен анализ постига с около 15 процентни пункта (около 30%) по-висока средна прецизност на извличане от VSM. При CISI и NPL, подобрението е с около 2 процентни пункта, което обаче е 10% разлика, защото и двата метода постигат доста ниска прецизност при тези колекции. При колекциите TIME и CACM, LSA се представя дори по-зле от VSM при брой на скритите теми под 400 за TIME и 2500 CACM, след което прецизността на двата метода се изравнява. Авторите дискутират какви биха били причините за тези толкова разнородни резултати. Moldovan [11] и Li [12] стигат до заключението, че вероятно върху точността на методите влияят както структурата на стандартизираните множества, така и самите данни (предметната област). Тук трябва да се отбележи, че стандартизираните множества от данни са вид „лабораторни данни“, които съдържат не само суровите данни, но и предварително подготвени заявки и списък с резултатите от всяка заявка, като са посочени кой резултат е адекватен и кой не. Но получената оценка не рядко е еднолична, предоставена от един експерт, еднократно и в определен контекст. В изследване на Dumais и Nielsen [13], експерт е помолен да повтори оценката си за адекватност на резултатите (същите резултати) около 7

месеца след първото му анкетиране, и се оказва, че вторият му отговор се различава значително от първия, като корелацията между двата отговора е 0.76.

Andreea Moldovan и съавтори [11] провеждат експериментален сравнителен анализ между LSA и VSM, приложени върху колекция от патентни документи от американското патентно ведомство за периода от 1790 г. до 2005 г. Анализът им показва, че резултатите от LSA и VSM всъщност са доста близки, като в по-голямата част от случите, LSA наистина постига по-добри резултати, но подобрението е с около 5%. Авторите цитират обаче и случаи, при които LSA се справя по-зле от VSM, като постига с 3% по-ниска прецизност на извличане на документите от VSM.

Проведените експериментални изследвания и сравнителни анализи също показват, че латентният семантичен анализ постига по-висока точност на изчислените подобия от векторния модел за анализ на текст. Когато матрицата „думи-документи“ е формирана само от честотата на поява на отделните думи в съответните документи, повишението в точността е само 2-3 процентни пункта (ppts), докато ако матрицата се състои от tf-idf теглата на думите, тогава точността се повишава средно с около 8-9 процентни пункта (което всъщност е над 10%). Тук е важно да се отбележи, че сравнението с VSM е направено при използване на може би най-добрия модел за изчисляване на теглата на думите при векторния модел – BM 25. Спрямо базовия tf-idf модел, увеличението е с малко над 10 процентни пункта.

Латентният семантичен анализ е надежден метод за изчисляване на подобията между отделните документи или между заявка и документите в колекцията. Преди обаче да бъде предпочетен пред други методи е желателно да се направи анализ на конкретната задача и евентуално да се прецени дали би могъл да бъде заменен от значително по-лесния за реализация векторен модел за анализ на текст. Времевата сложност на LSA от $O(n^3)$, предвид очакваните огромни стойности на n от десетки хиляди, го прави не много подходящ за работа в реално време при големи колекции от документи. При колекция от около 5 000 документи, броят на уникалните думи в нея е около и над 20 000, което определя и стойността на n. Но за разлика от VSM, тук няма възможност за реализация чрез инвертиран индекс, при което да отпаднат по-голямата част от изчисленията.

При подготовката на този файл са използвани материали от:

Калмуков, Й. Методи и алгоритми за търсене и извличане на документи. Издателство Primax Русе, 2022 г., ISBN 978-619-7242-93-5

Литература

1. Deerwester S., Dumais S.T., Furnas G.W., Landauer T.K. and Harshman R. (1990): Indexing by latent semantic analysis. —J. Amer. Soc. Inf. Sci., Vol. 41, No. 6, pp. 391–407.
2. Schwarzenberg-Czerny, A. (1995). "On matrix factorization and efficient least squares solution". Astronomy and Astrophysics Supplement Series. 110: 405.
3. Bareiss, Erwin H. (1968), "Sylvester's Identity and multistep integer-preserving Gaussian elimination", Mathematics of Computation, 22 (103): 565–578, doi:10.2307/2004533, JSTOR 2004533

4. Bunch, J. R.; Hopcroft, J. E. (1974). "Triangular Factorization and Inversion by Fast Matrix Multiplication". Mathematics of Computation. 28 (125): 231–236. doi:10.1090/S0025-5718-1974-0331751-8.
5. Aho, Alfred V., Hopcroft, John E., Ullman, Jeffrey D. (1974), The Design and Analysis of Computer Algorithms, Addison-Wesley, Theorem 6.6, p. 241
6. Francis, J.G.F. "The QR Transformation, I", The Computer Journal, 4(3), pages 265–271 (1961). doi:10.1093/comjnl/4.3.265
7. Francis, J. G. F. (1962). "The QR Transformation, II". The Computer Journal. 4 (4): 332–345. doi:10.1093/comjnl/4.4.332
8. Vera N. Kublanovskaya, "On some algorithms for the solution of the complete eigenvalue problem," USSR Computational Mathematics and Mathematical Physics, vol. 1, no. 3, pages 637–657 (1963). doi:10.1016/0041-5553(63)90168-X
9. Golub, G.H.; van der Vorst, H.A. (2000). "Eigenvalue computation in the 20th century". Journal of Computational and Applied Mathematics. 123 (1–2): 35–65. doi:10.1016/S0377-0427(00)00413-1
10. Golub, Gene H.; Kahan, William (1965). "Calculating the singular values and pseudo-inverse of a matrix". Journal of the Society for Industrial and Applied Mathematics, Series B: Numerical Analysis. 2 (2): 205–224. Bibcode:1965SJNA....2..205G. doi:10.1137/0702016. JSTOR 2949777.
11. Moldovan, Andreea, Radu Ioan Bot, and Gert Wanka. "Latent semantic indexing for patent documents." (2005).
12. Li, Dandan, and Chung-Ping Kwong. "Understanding latent semantic indexing: A topological structure analysis using Q-analysis." Journal of the american society for information science and technology 61, no. 3 (2010): 592-608.
13. Dumais, Susan T., and Jakob Nielsen. "Automating the assignment of submitted manuscripts to reviewers." In Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval, pp. 233-244. 1992.