Latent Semantic Indexing: Einführung und Experiment

Jonathan Geiger, Felix Hieber

HS: Information Retrieval
Dr. Haenelt

12.01.2009
WS 08/09

Motivation

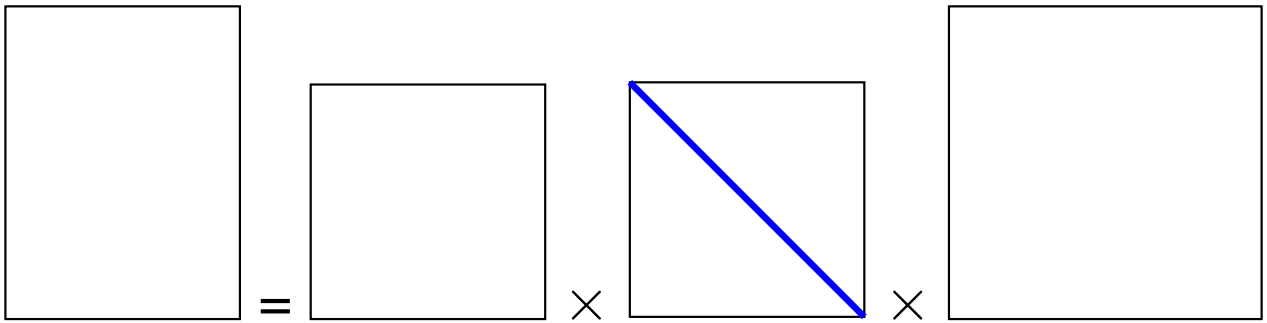
Grundsätzlich stecken zwei Ideen hinter LSI, eine praktischer und eine theoretischer Natur:

- Dimensionsreduktion von Matrizen verringern den Speicherplatzbedarf
- Durch die Dimensionsreduktionen landen im idealen Fall ähnliche Terme in derselben Dimension.

Was passiert bei LSI ?

- Eine Dokument-Term-Matrix wird dimensionsreduziert;
- Sie wird durch *Singulärwertzerlegung* in drei Matrizen T, S, D zerlegt
- Aus der Diagonalmatrix S werden Dimensionen mit einem Wert unter einem bestimmten Schwellwert entfernt $\longrightarrow S'$
- Entsprechende Reihen/Spalten werden aus T und D entfernt $\longrightarrow T', D'$
- Mittels T', S' und D' werden dann die Ähnlichkeiten zwischen Dokumentvektoren berechnet

Aufteilung der Dokument-Term-Matrix



SVD anhand eines Beispiels

Term-Dokument-Matrix

	d1	d2	d3	q
Auto	5	0	3	1
Wagen	0	4	4	0
Mississippi	2	3	1	0

mathematisch

$$\begin{pmatrix} 5 & 0 & 3 & 1 \\ 0 & 4 & 4 & 0 \\ 2 & 3 & 1 & 0 \end{pmatrix}$$

Aufteilung der Dokument-Term-Matrix

- Wenn keine quadratische Matrix vorliegt (und das tut sie selten), werden zwei quadratische Matrizen zur Berechnung von T und D gebildet:
  $X = A \cdot A'$
  $Y = A' \cdot A$
- In unserem Fall:

$$\begin{pmatrix} 5 & 0 & 3 & 1 \\ 0 & 4 & 4 & 0 \\ 2 & 3 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 5 & 0 & 2 \\ 0 & 4 & 3 \\ 3 & 4 & 1 \\ 1 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 35 & 12 & 13 \\ 12 & 32 & 16 \\ 13 & 16 & 14 \end{pmatrix} = X$$

$$\begin{pmatrix} 5 & 0 & 2 \\ 0 & 4 & 3 \\ 3 & 4 & 1 \\ 1 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} 5 & 0 & 3 & 1 \\ 0 & 4 & 4 & 0 \\ 2 & 3 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 29 & 6 & 17 & 5 \\ 6 & 25 & 19 & 0 \\ 17 & 19 & 26 & 3 \\ 5 & 0 & 3 & 1 \end{pmatrix} = Y$$

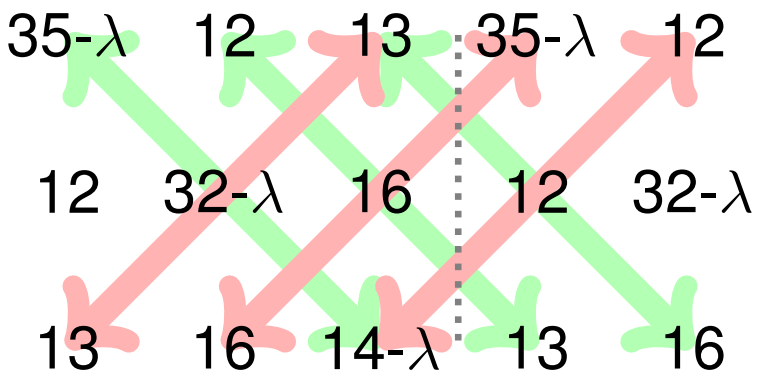
Schritt 1: Finden der Eigenwerte I

- exemplarisch nur für X
- Auflösen der Gleichung $det\,|A - \lambda E| = 0$ nach λ
- Zunächst: $E = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

Schritt 1: Finden der Eigenwerte II

- $det \left| \begin{pmatrix} 35 & 12 & 13 \\ 12 & 32 & 16 \\ 13 & 16 & 14 \end{pmatrix} - \lambda \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \right| = 0$
- $det \left| \begin{pmatrix} 35 & 12 & 13 \\ 12 & 32 & 16 \\ 13 & 16 & 14 \end{pmatrix} - \begin{pmatrix} \lambda & 0 & 0 \\ 0 & \lambda & 0 \\ 0 & 0 & \lambda \end{pmatrix} \right| = 0$
- $det \left| \begin{pmatrix} 35-\lambda & 12 & 13 \\ 12 & 32-\lambda & 16 \\ 13 & 16 & 14-\lambda \end{pmatrix} \right| = 0$

Determinante berechnen



$$det\,|A| = (35-\lambda)\cdot(32-\lambda)\cdot(14-\lambda) + 12\cdot 16\cdot 13 + 13\cdot 12\cdot 16 - 13\cdot(32-\lambda)\cdot 13 - (35-\lambda)\cdot 16\cdot 16 - 12\cdot 12\cdot(14-\lambda)$$

$$\ldots$$

$$= -\lambda^3 + 81\lambda^2 - 1489\lambda + 4288$$

Jetzt noch die Gleichung lösen...

$$\lambda^3 - 81\lambda^2 + 1489\lambda - 4288 = 0$$

...und der Größe nach ordnen

1. $\lambda_1 = 55.6114689$
2. $\lambda_2 = 21.8614921$
3. $\lambda_3 = 3.52704202$

Eigenvektoren bauen: Gleichung

Die zu lösende Gleichung

$$(A - \lambda) \cdot \vec{x} = 0$$

Die zu lösende Gleichung: eingesetzt

$$\begin{pmatrix} 35-\lambda & 12 & 13 \\ 12 & 32-\lambda & 16 \\ 13 & 16 & 14-\lambda \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = 0$$

als Gleichungssystem

$$\begin{array}{llll} (35-\lambda)\cdot x_1 & +12\cdot x_2 & +13\cdot x_3 & = 0 \\ 12\cdot x_1 & +(32-\lambda)\cdot x_2 & +16\cdot x_3 & = 0 \\ 13\cdot x_1 & +16\cdot x_2 & +(14-\lambda)\cdot x_3 & = 0 \end{array}$$

Eigenvektoren bauen: Einsetzen

- Es ergibt sich:

$$\begin{aligned} x_1 &= \alpha \\ x_2 &= \frac{-404+16\lambda}{-224+13\lambda}\alpha \\ x_3 &= \frac{-404+16\lambda}{40+12\lambda}\alpha \end{aligned}$$

- Nun setzt man jeden Eigenwert ein und normiert den entstehenden Vektor.

$\lambda_1 = 55.6114689$

$$\begin{aligned} x_1 &= \alpha \\ x_2 &= 0.9736\alpha \\ x_3 &= 0.6868\alpha \end{aligned}$$

$\lambda_2 = 21.8614921$

$$\begin{aligned} x_1 &= \alpha \\ x_2 &= -0.9001\alpha \\ x_3 &= -0.1794\alpha \end{aligned}$$

$\lambda_3 = 3.52704202$

$$\begin{aligned} x_1 &= \alpha \\ x_2 &= 1.9510\alpha \\ x_3 &= -4.2220\alpha \end{aligned}$$

Eigenvektoren bauen: Normierung

- bisher nur unnormierter Eigenvektor: $s = \alpha \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$
- Normierung: $\sqrt{s' \cdot s} = 1$ nach α auflösen.
- Beispiel: der durch λ_1 erhaltene Vektor.
- $$\sqrt{(\alpha \;\; 0.9736\alpha \;\; 0.6868\alpha) \begin{pmatrix} \alpha \\ 0.9736\alpha \\ 0.6868\alpha \end{pmatrix}} = 1.$$
- $\sqrt{\alpha^2 + 0.9479\alpha^2 + 0.4717\alpha^2} = 1$
- $\sqrt{2.4196\alpha^2} = 1$
- $\alpha = 0.6428$
- Es ergibt sich der Eigenvektor $\begin{pmatrix} 0.6428 \\ 0.6259 \\ 0.4415 \end{pmatrix}$

Eigenvektoren bauen: Ergebnisse I

- Für λ_2 erhält man:
$$\begin{pmatrix} 0.7366 \\ -0.6634 \\ -0.1321 \end{pmatrix}$$
- Für λ_3 schließlich:
$$\begin{pmatrix} -0.2102 \\ -0.4101 \\ 0.8875 \end{pmatrix}$$

Eigenvektoren bauen: Ergebnisse II

Die Eigenwerte für Y lauten

$\lambda_1 = 56.11$

$\lambda_2 = 21.86$

$\lambda_3 = 3.52$

$\lambda_4 = 0$

Die zugehörigen Eigenvektoren

$$\begin{pmatrix} 0.549451 \\ 0.513349 \\ 0.653562 \\ 0.086207 \end{pmatrix} \begin{pmatrix} 0.731159 \\ -0.65224 \\ -0.123151 \\ 0.157531 \end{pmatrix} \begin{pmatrix} -0.385465 \\ -0.54417 \\ 0.736725 \\ 0.111929 \end{pmatrix} \begin{pmatrix} -0.122169 \\ 0.122169 \\ -0.1221694 \\ 0.977355 \end{pmatrix}$$

T S D: Erklärung

Aus den gewonnenen Eigenvektoren baut man nun T, S und D:

1. In T bestehen die Spalten aus den Eigenvektoren (von X)
2. In S stehen die nach Größe abwärts geordneten Singulärwerte auf der Hauptdiagonalen
3. Singulärwerte=Wurzeln der Schnittmenge der Eigenwerte von X und Y
4. In D bestehen die Reihen aus den Eigenvektoren (von Y)

T S D: Werte

$$T \begin{pmatrix} 0.642879 & 0.736558 & -0.210207 \\ 0.625915 & -0.663351 & -0.41011 \\ 0.441514 & -0.132081 & 0.887478 \end{pmatrix}$$

$$S \begin{pmatrix} \sqrt{55.611465} & 0 & 0 \\ 0 & \sqrt{21.86149} & 0 \\ 0 & 0 & \sqrt{3.527042} \end{pmatrix}$$

$$D \begin{pmatrix} 0.549451 & 0.731159 & -0.385465 & -0.122169 \\ 0.513349 & -0.6522445 & -0.5441725 & 0.122169 \\ 0.653562 & -0.123151 & 0.736725 & -0.122169 \\ 0.086207 & 0.157531 & 0.111929 & 0.977355 \end{pmatrix}$$

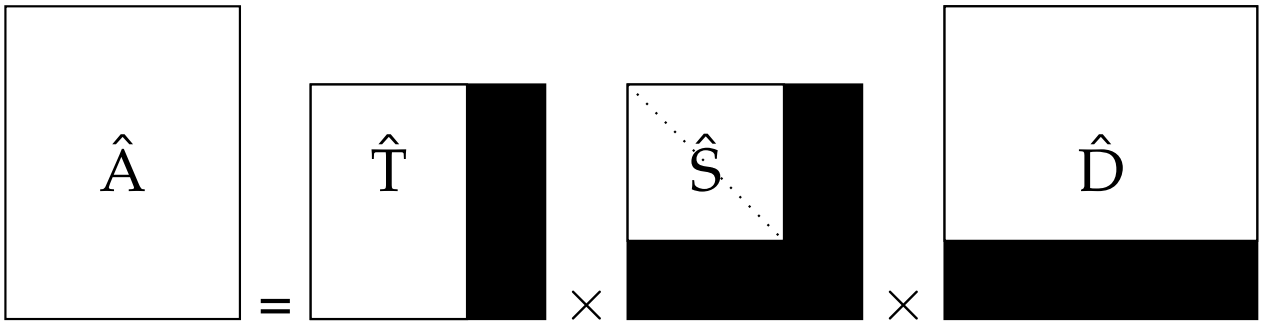
Reduktion: Hintergrund I

- Betrachten wir die Singulärwertmatrix M= $\begin{pmatrix} 60 & 0 & 0 \\ 0 & 71 & 0 \\ 0 & 0 & 2 \end{pmatrix}$

  und den Vektor $\begin{pmatrix} 2 \\ 4 \\ 6 \end{pmatrix}$
- Ihr Produkt lautet $\begin{pmatrix} 120 \\ 284 \\ 12 \end{pmatrix}$
- Man erkennt, dass der Einfluss des letzten Singulärwerts aus M so gering ist, dass man ihn ohne großen Schaden entfernen könnte.
- Damit verringerte sich die Dimension der Matrix um 1.
- Dieses Prinzip wollen wir auch auf *S* anwenden.

Reduktion: Hintergrund II

- Wir legen einen Schwellwert k fest.
- Alle Werte aus S, die unter dem Schwellwert liegen, werden samt ihrer Dimension entfernt.
- Die zugehörigen Spalten und Reihen aus T und D werden ebenfalls entfernt.



Reduktion bei unserem Beispiel

- Wir entscheiden uns für den Schwellwert k=2 ($> \sqrt{3.52}$).

T' $\begin{pmatrix} 0.642879 & 0.736558 \\ 0.625915 & -0.663351 \\ 0.441514 & -0.132081 \end{pmatrix}$

S' $\begin{pmatrix} \sqrt{55.611465} & 0 \\ 0 & \sqrt{21.86149} \end{pmatrix}$

D' $\begin{pmatrix} 0.549451 & 0.731159 & -0.385465 & -0.122169 \\ 0.513349 & -0.6522445 & -0.5441725 & 0.122169 \\ 0.653562 & -0.123151 & 0.736725 & -0.122169 \end{pmatrix}$

Zugriff auf Dokumente

- Um auf die neuen Dokumentvektoren zu kommen, darf nicht das Produkt von T', S', D' zurückgegriffen werden.
- Vielmehr benutzt man die Gleichung $d = d^T T' S'^{-1}$
- Damit erhält man einen neuen Vektor, der wie üblich mit dem Anfragevektor verglichen werden kann (die Anfrage wird auch als Dokument behandelt)

Zusammenfassung

1. Dokument-Term-Matrix erstellen
2. Matrix in zwei Hilfsmatrien unterteilen
3. Eigenwerte und -vektoren finden
4. T', S', D' erstellen
5. mittels obiger Gleichung auf Dokumentvektoren zugreifen
6. Ähnlichkeiten berechnen und ausgeben

Die Praxis unseres Systems

Wir haben einen Ansatz mit LSI implementiert:

- Python-Modul NUMPY
  - kann mit Matrizen rechnen ...
  - und kann sogar die Eigenwertzerlegung! (`numpy.linalg.svd`)
- Stoppwortfilterung
- Cosinus als Ähnlichkeitsmaß
- Schwellwert optional als Konstante oder relativ einstellbar

Ein Beispiel

- Wir nehmen uns zwanzig Dateien
- Zehn Stück enthalten Schachbegriffe
- Die anderen zehn bestehen aus Begriffen aus dem Wortfeld ’Auto’
- Sie bestehen aus jeweils 50-100 Wörtern.
- Die beiden Datensätze haben keinen Begriff gemeinsam

Ergebnisse ohne LSI

- Schauen wir uns zunächst die Ergebnisse an, wenn wir das normale Vektormodell benutzen, wenn wir als Anfrage *auto, autobahn, kofferraum* stellen:

auto_6	0.4639
auto_1	0.4233
auto_3	0.4139
...	
auto_2	0.3379
schach_0-9	0.0

Ergebnisse mit LSI

- Vergleichen wir also, was wir mit LSI herausbekommen (k=80%)

auto_3	0.5531
auto_6	0.5229
auto_1	0.4883
auto_5	0.125
auto_0	0.0086
schach_0-9	0.0
auto_7	-0.0746
auto_4	-0.093
auto_9	-0.1862
auto_2	-0.3932
auto_8	-0.435

- Weiterhin bekommen die Schachdateien also den Wert 0, allerdings werden einige Dateien mit negativen Werten belegt (was im Vektormodell nicht geht)

Andere Daten-gleiche Ergebnisse

- wir fügen eine Datei hinzu, die *sowohl* Schach-, *als auch* Autobegriffe enthält.
  - Im normalen Vektormodell ändern sich die Ergebnisse nicht:

auto_6	0.4639
auto_1	0.4233
auto_3	0.4139
auto_4	0.4105
auto_2	0.3379
schachauto	0.3044
schach_0-9	0.0

Andere Daten-andere Ergebnisse

- Benutzen wir LSI, sehen die Ergebnisse aber deutlich anders aus:

auto_3	0.5861	schach_8	0.003
auto_6	0.5292	schach_5	0.0003
auto_1	0.4411	schach_9	-0.0134
schachauto	0.0904	schach_3	-0.0336
auto_0	0.0732	auto_7	-0.0887
schach_2	0.0641	auto_4	-0.1125
schach_6	0.0529	auto_9	-0.167
schach_7	0.0445	schach_1	-0.2786
auto_5	0.0417	auto_2	-0.3455
schach_0	0.0331	auto_8	-0.439
schach_4	0.0322		

- Wir sehen also, dass nun auch die Schachdateien mehr oder weniger hohe Ähnlichkeiten aufweisen

Termähnlichkeiten

- Unser Programm bietet außerdem die Möglichkeit, die Ähnlichkeit zwischen den Termen auszugeben
- Schauen wir uns die folgenden Dateien an:

Datei_1 Auto Auto Auto Auto Auto Mississippi Mississippi

Datei_2 Wagen Wagen Wagen Wagen Mississippi Mississippi Mississippi

Datei_3 kashikoshi

##########	auto	mississippi	wagen	kashikoshi
auto	##########	0.6535743	0.2228344	0.0000000
mississippi	##########	##########	0.8834709	0.0000000
wagen	##########	##########	##########	0.0000000

Fazit

- *Everything is pretty much straightforward and crystal clear.*[1]
- Idee: Durch Dimensionsreduktion fallen ähnliche Terme (hoffentlich) in gleiche Dimension
- **kristallklare** Mathematik
- . . . die man nicht implementieren muss :)
- Kleine Tests zeigen, dass der Ansatz erfolgsversprechend sein kann

[1] http://www.miislita.com/information-retrieval-tutorial/svd-lsi-tutorial-4-lsi-how-to-calculations.html

Referenzen

- Dr. E. Garcia, *A tutorial on Singular Value Decomposition (SVD) and Latent Semantic Indexing (LSI), its advantages, applications and limitations. Covers LSI myths and misconceptions from search engine marketers. Juli 2007* http://www.miislita.com/information-retrieval-tutorial/svd-lsi-tutorial-1-understanding.html
- Johanna Geiß, Jette Klein-Berning, Tutorial: Latent Semantic Indexing, Februar 2003
- Manning/Schütze, Introduction to Information Retrieval, Cambridge University Press 2008, *bes. Kapitel 18 siehe* http://www-csli.stanford.edu/∼hinrich/information-retrieval-book.html