FFHS
Fernfachhochschule Schweiz
Zürich | Basel | Bern | Brig
Mitglied der SUPSI

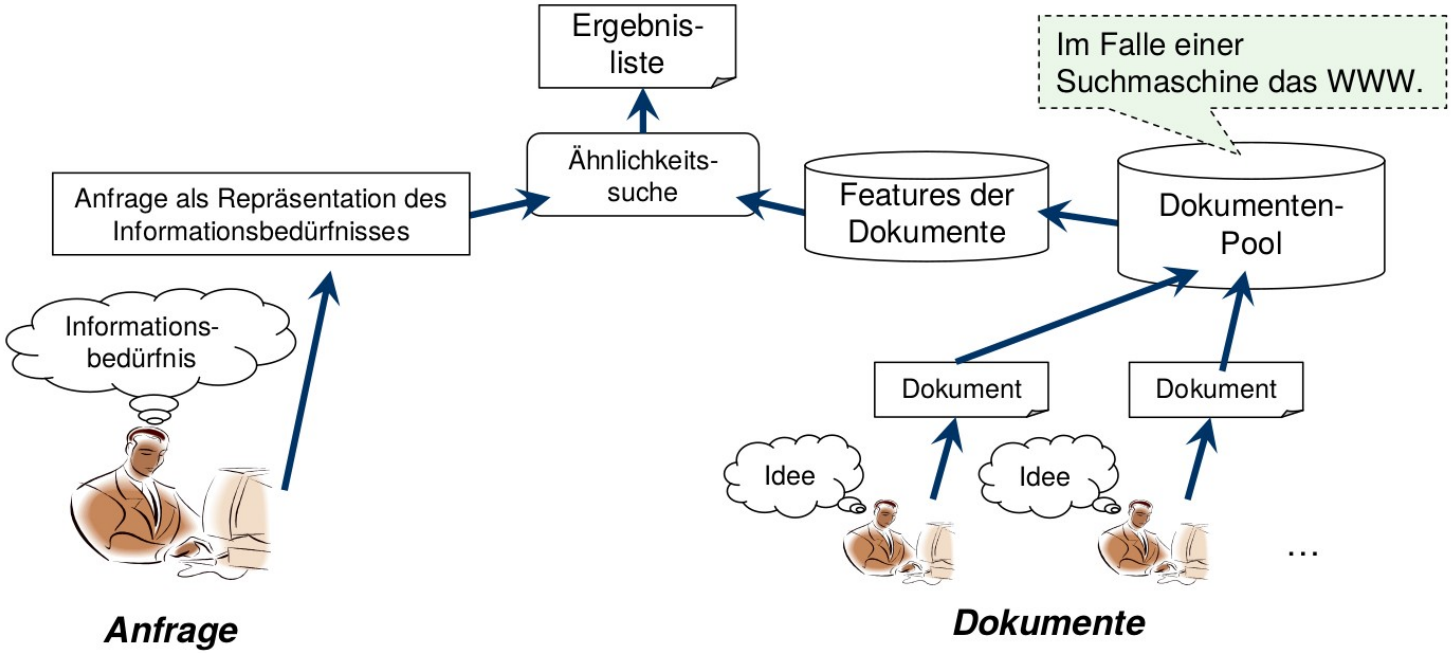
Einführung in Data Science – Block5
Information Retrieval

Programm

Thema	Form	Zeit
Besprechung der Semesterarbeit und Hausaufgaben	Besprechung	13:45 – 14:00
Vorlesung	Vorlesung	14:00 – 14:45
Pause+Modulevaluation		14:45 – 15:15
Musterprüfung	Workshop	15:15 – 16:15
Besprechung Musterprüfung	Diskussion	16:15 – 17:00

Information Retrieval

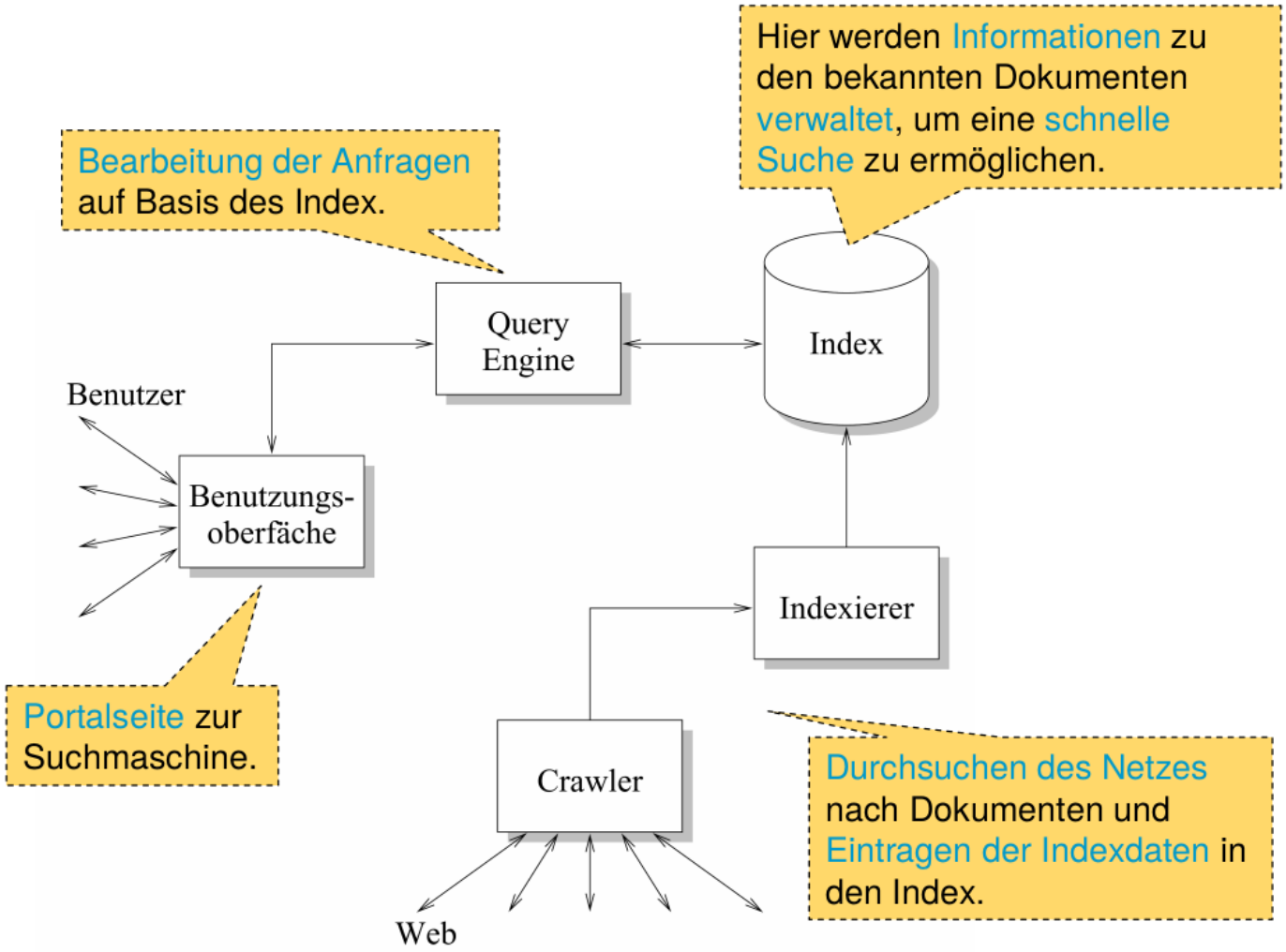
- Bei einer Sammlung von Dokumenten und einem Informationsbedürfnis des Benutzers besteht der Zweck eines IR-Systems darin, Informationen zu finden, die für den Benutzer nützlich oder relevant sein könnten.



Funktionalität von Suchmaschinen

- Suche
- Katalog
- …

Was steckt dahinter?



Information Retrieval
Web Search Probleme

- Verteilte Daten
- Flüchtige und variable Daten
- Riesige Datenmengen
- Unstrukturierte und redundante Daten
- Datenqualität
- Heterogene Daten

Ansätze zum IR

- **Klassifikation**
  - Dokumente werden (manuell) in ein Klassifikationsschema eingeordnet.
- **Anfrage**
  - Der Benutzer formuliert seinen “Informationswunsch” als Anfrage.
  - Das System versucht (automatisch) hierzu relevante Dokumente zu finden.

Ansätze zum IR

- **Browsing**
  - Der Benutzer will die Dokumentenkollektion interaktiv erarbeiten.
- **Information Filtering**
  - Aus einem andauernden Strom von Dokumenten sollen automatisch die den Benutzer interessierenden ermittelt werden.

Text Mining/Text Analytics

- Die Entdeckung neuer, bisher unbekannter Informationen durch einen Computer durch die automatische Extraktion verschiedener schriftlicher (unstrukturierter) Dokumente.
- Beispiele:
  - Sentiment Analysis
  - Inhaltsanalyse
  - Automatisches Clustering und automatische Kategorisierung
  - Bedeutung der Texte automatisch erkennen

Dokument/Retrieval Unit

- Web pages, email, books, news, stories, scholarly papers, text messages, Word, Power Point, PDF, forum postings, patents, etc.

- Retrieval Unit können sein:
  - Teil eines Dokuments, z.B. einen Abschnitt, eine Folie, eine Seite, etc.
  - Verschiedene strukturen: html, xml, text, etc.
  - Verschiedene Länge (Sizes)

Text Mining Applikationen

- Information Retrieval
- Textklassifizierung
- Textclustering
- Informationsextraktion

Schritte im IR

- Informationsbedarf kennen
- Daten beschaffen
- Daten kennen lernen
- Daten konvertieren
- Daten indexieren
- Daten interpretieren

Informationsbedarf/ Datenbeschaffung

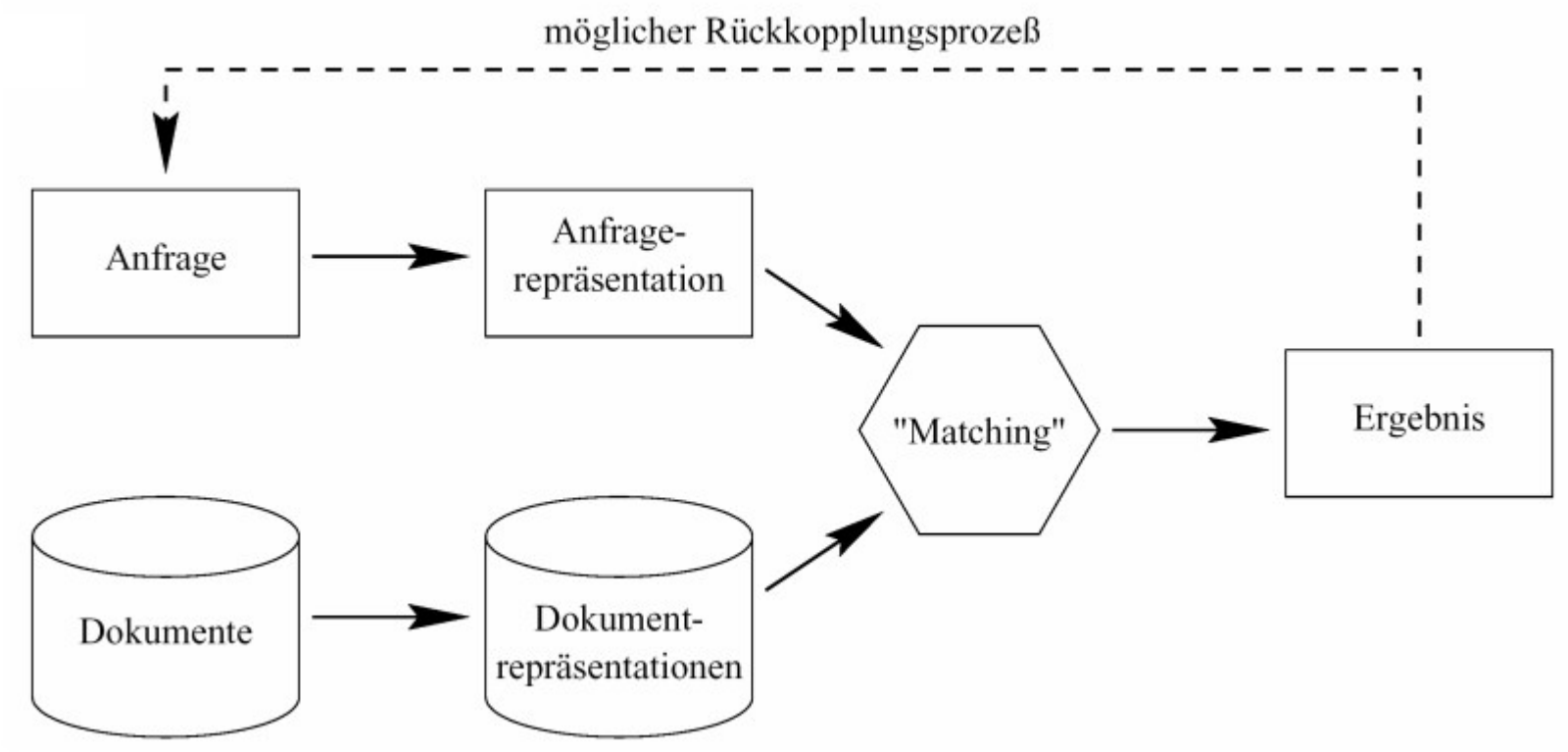
- Komponenten einer Suchmaschine:
  - Web-Robot-System: zur Erfassung von neuen und veränderten Daten
  - Information Retrieval System zur Aufbereitung und Bewertung der erfassten Daten
  - Query Processor liefert zur Suchanfrage passende Ergebnisse aus einer Datenbank

- Query Beispiele:
- Einfache Abfrage
  - Einige Keywords oder mehr
- Boolesche Abfrage
    - 'neuronales Netzwerk AND Spracherkennung'
- Spezielle Abfrage
  - 400 chf in usd

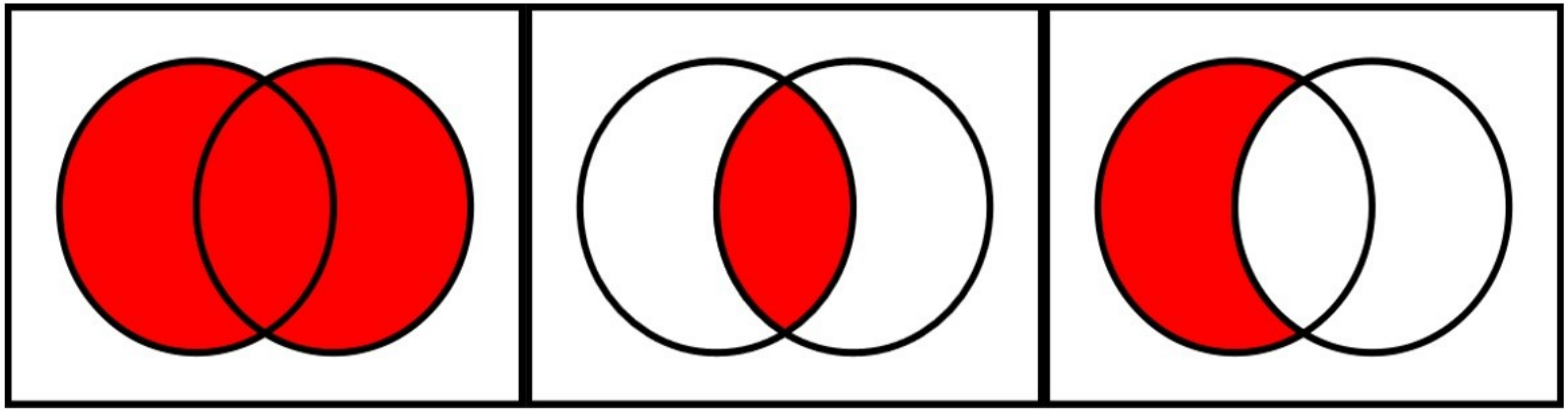
IR Models

- **Das Boolesche IR-Modell**
- **Vector space model**
- Statistical language model
- Etc.

Allgemeines IR-Modell

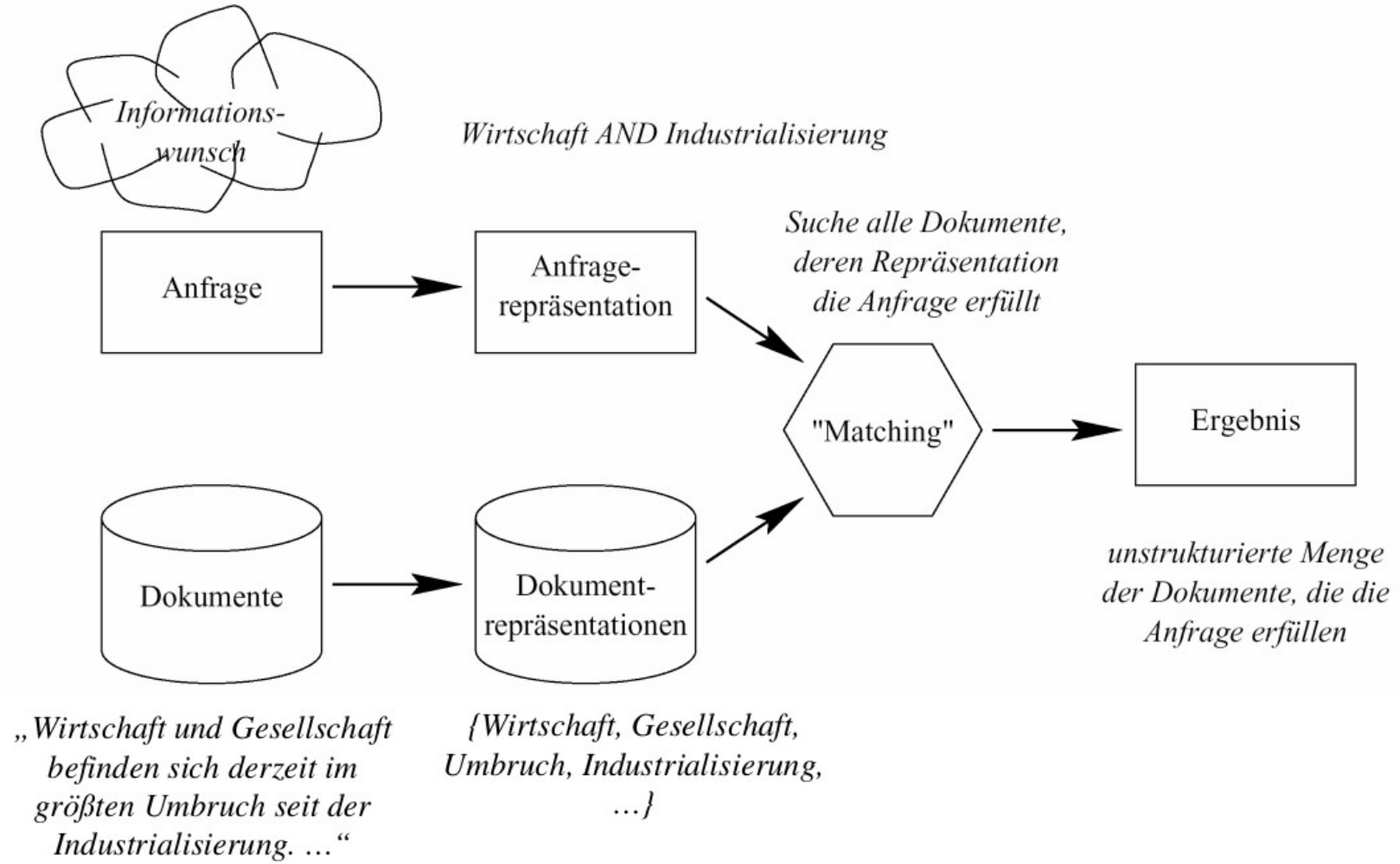


Das Boolesche IR-Modell



OR AND NOT

Das Boolesche IR-Modell



Nachteile des Booleschen Retrieval

- Keine Rückführung der Wörter auf eine Grundform
- Keine Gewichtung der Wörter (weder nach Ort noch nach Häufigkeit)
- Keine Zerlegung von Mehrwortgruppen
- Relativ aufwendige Formulierung der Anfrage
- Die Ergebnismenge ist unstrukturiert
- Kein Ranking der Dokumente

Beispiel

- Welche Theaterstücke von Shakespeare enthalten die Wörter:
  - **Brutus AND Caesar NOT Calpurnia**?
  - → grep alle Theaterstücke nach ***Brutus*** und ***Caesar***, dann die Zeilen streichen, die ***Calpurnia*** enthalten?
  - Langsam (für grosse Kollektionen)
  - **NOT Calpurnia** ist nicht trivial
  - Ranking nicht möglich

Term-Dokument Matrix

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	1	1	0	0	0	1
Brutus	1	1	0	1	0	0
Caesar	1	1	0	1	1	1
Calpurnia	0	1	0	0	0	0
Cleopatra	1	0	0	0	0	0
mercy	1	0	1	1	1	1
worser	1	0	1	1	1	0
...						

Term-Dokument Matrix

QUERY: Brutus AND Caesar NOT Calpurnia

110100 (Brutus)

110111 (Caesar)

010000(Calpurnia)

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	1	1	0	0	0	1
Brutus	1	1	0	1	0	0
Caesar	1	1	0	1	1	1
Calpurnia	0	1	0	0	0	0
Cleopatra	1	0	0	0	0	0
mercy	1	0	1	1	1	1
worser	1	0	1	1	1	0
...						

Term-Dokument Matrix

QUERY: Brutus AND Caesar NOT Calpurnia → Komplementär von Calpurnia: 101111

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	1	1	0	0	0	1
Brutus	1	1	0	1	0	0
Caesar	1	1	0	1	1	1
¬Calpurnia	1	0	1	1	1	1
Cleopatra	1	0	0	0	0	0
mercy	1	0	1	1	1	1
worser	1	0	1	1	1	0
...						

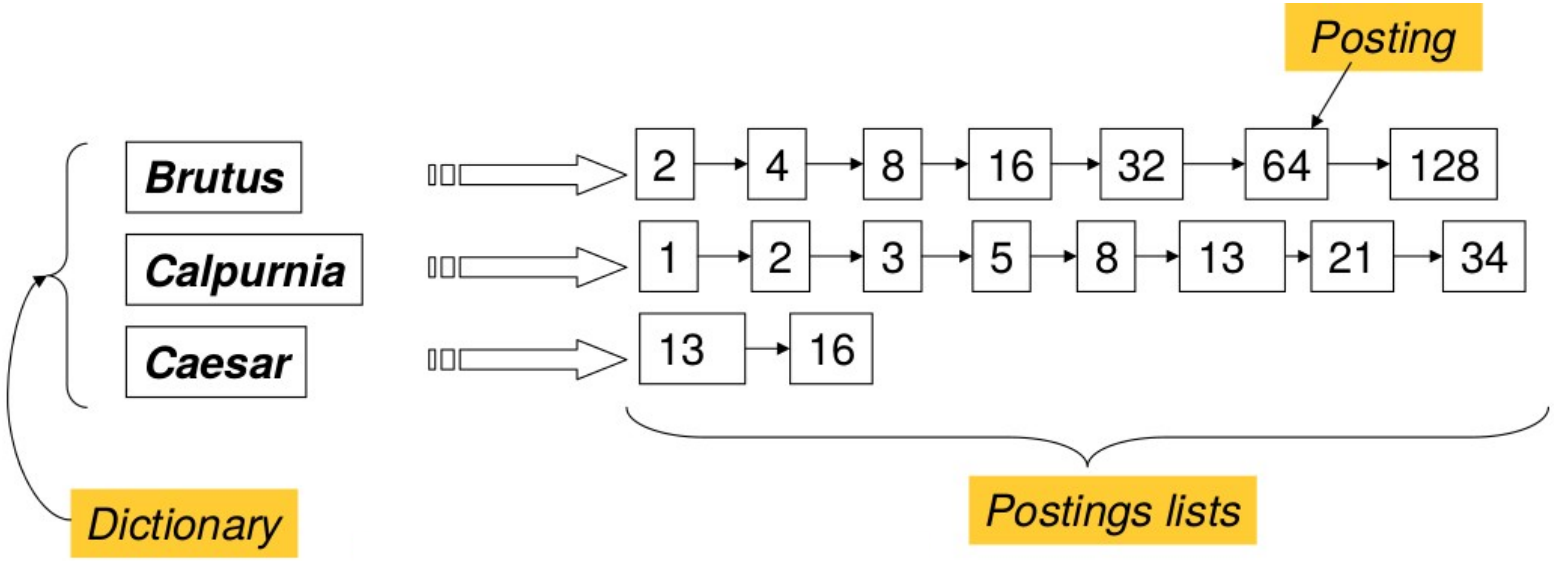
Term-Dokument Matrix

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	1	1	0	0	0	1
Brutus	1	1	0	1	0	0
Caesar	1	1	0	1	1	1
¬Calpurnia	1	0	1	1	1	1
Cleopatra	1	0	0	0	0	0
mercy	1	0	1	1	1	1
worser	1	0	1	1	1	0
AND	1	0	0	1	0	0

Problem bei grossen Kollektionen

- Annahmen:
  - N = 10^6, 1000 Tokens je (2-3 Seiten)
  - 10^9 Tokens, Avg. 6 bytes/Wort → 6GB Datenvolumen der Dokumente
- Weitere Annahme:
  - M = 500k verschiedene Wörter (Vokabular von 500K Wörter)
  - 500k x 10^6 matrix → halb Miliarde 0 und 1, Sparse (dünn besetzt)
  - Bessere Representation? Wie registrieren nur die 1
  - Dies führt uns zu **invertierten Listen**

Invertierter Index



Verarbeitungsschritte des Indexers

- Sequenz von Paaren <Token, Dok. ID>.

Dokument 1

I did enact Julius Caesar I was killed i' the Capitol; Brutus killed me.

Dokument 2

So let it be with Caesar. The noble Brutus hath told you Caesar was ambitious

Term	Doc #
I	1
did	1
enact	1
julius	1
caesar	1
I	1
was	1
killed	1
i'	1
the	1
capitol	1
brutus	1
killed	1
me	1
so	2
let	2
it	2
be	2
with	2
caesar	2
the	2
noble	2
brutus	2
hath	2
told	2
you	2
caesar	2
was	2
ambitious	2

Verarbeitungsschritte des Indexers

- Sortiere Terme alphabetisch.

Term	Doc #
I	1
did	1
enact	1
julius	1
caesar	1
I	1
was	1
killed	1
i'	1
the	1
capitol	1
brutus	1
killed	1
me	1
so	2
let	2
it	2
be	2
with	2
caesar	2
the	2
noble	2
brutus	2
hath	2
told	2
you	2
caesar	2
was	2
ambitious	2

Term	Doc #
ambitious	2
be	2
brutus	1
brutus	2
capitol	1
caesar	1
caesar	2
caesar	2
did	1
enact	1
hath	1
I	1
I	1
i'	1
it	2
julius	1
killed	1
killed	1
let	2
me	1
noble	2
so	2
the	1
the	2
told	2
you	2
was	1
was	2
with	2

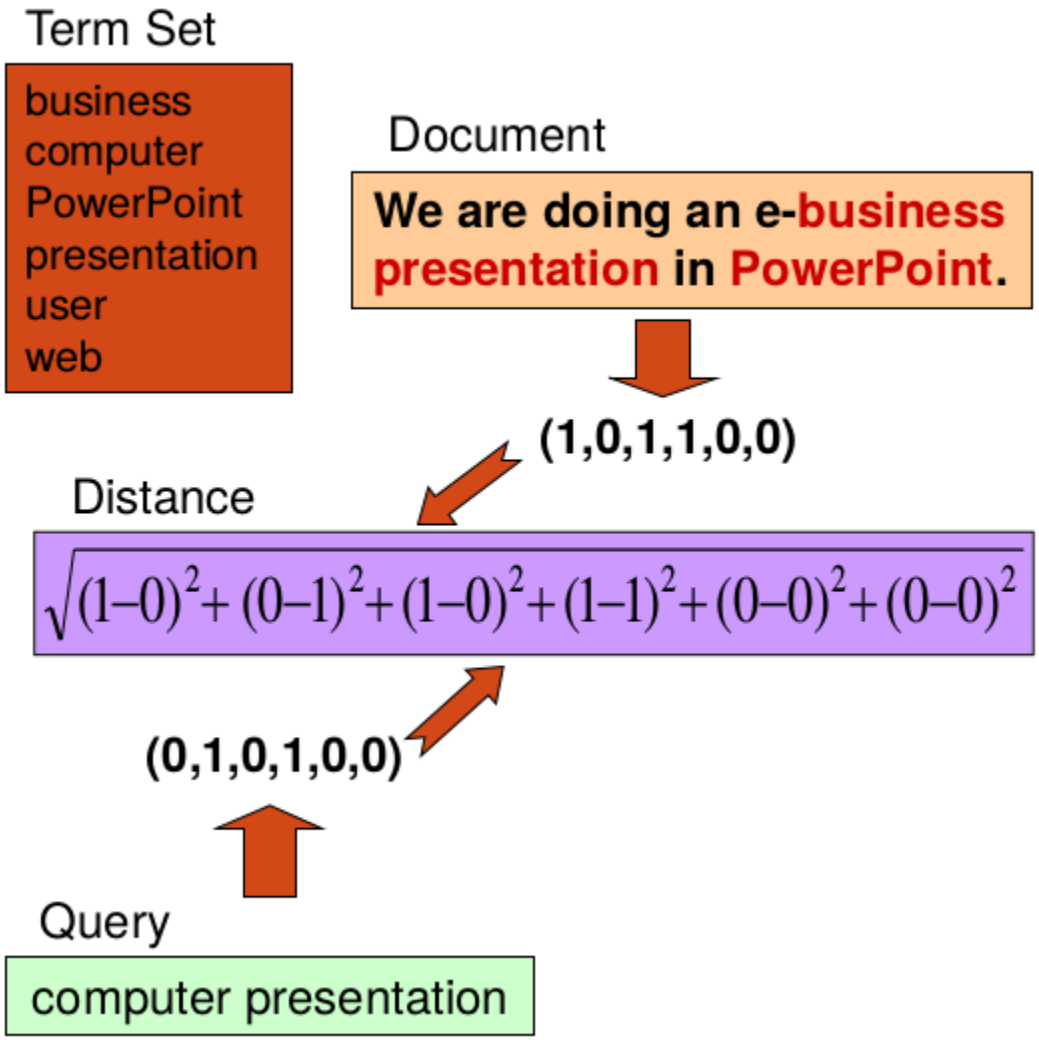
Verarbeitungsschritte des Indexers

- Mehrfacheinträge von Termen im gleichen Dokument werden zusammengeführt.
- Vorkommenshäufigkeit wird hinzugefügt.

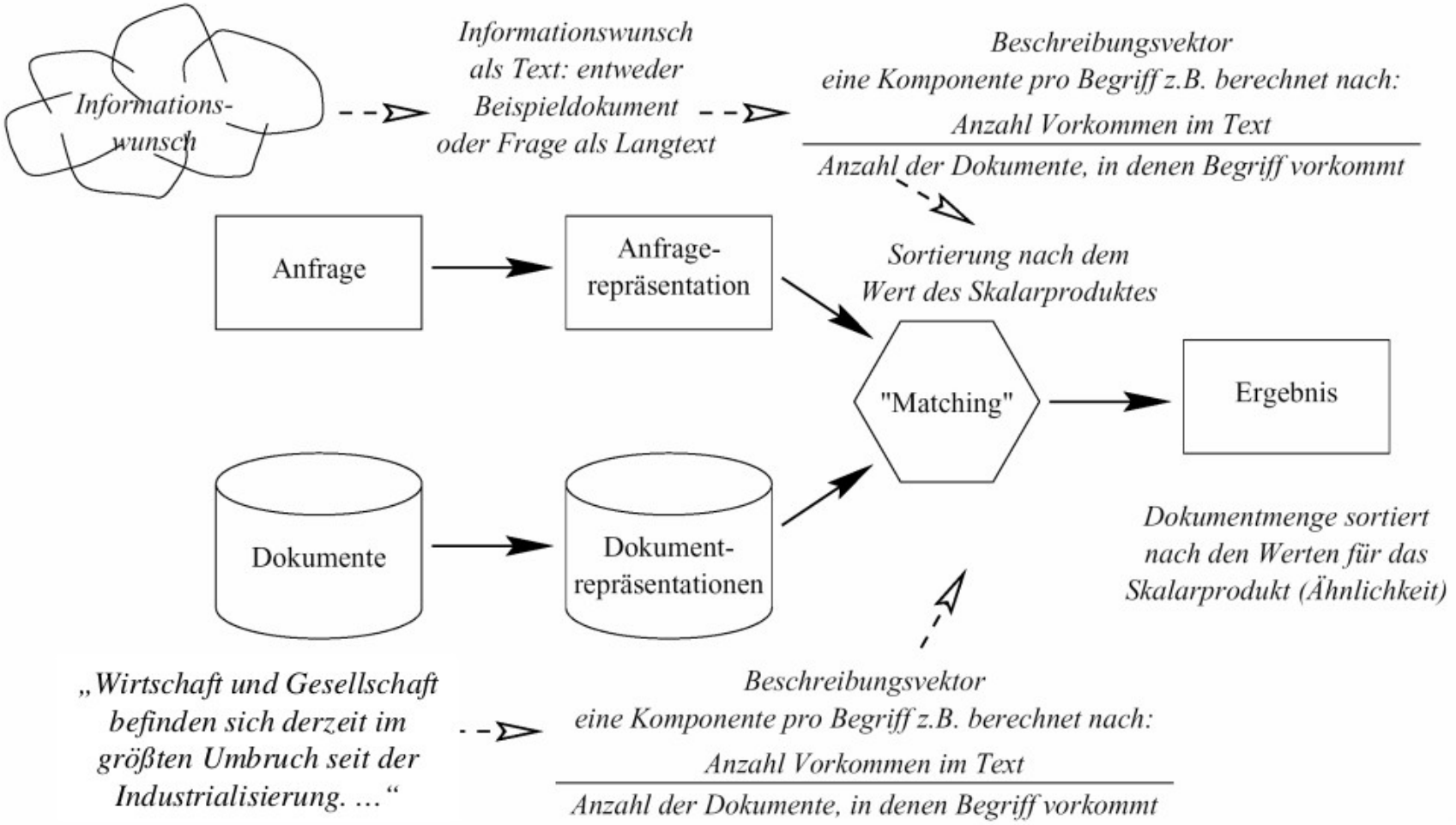
Term	Doc #
ambitious	2
be	2
brutus	1
brutus	2
capitol	1
caesar	1
caesar	2
caesar	2
did	1
enact	1
hath	1
I	1
I	1
i'	1
it	2
julius	1
killed	1
killed	1
let	2
me	1
noble	2
so	2
the	1
the	2
told	2
you	2
was	1
was	2
with	2

Term	Doc #	Term freq
ambitious	2	1
be	2	1
brutus	1	1
brutus	2	1
capitol	1	1
caesar	1	1
caesar	2	2
did	1	1
enact	1	1
hath	2	1
I	1	2
i'	1	1
it	2	1
julius	1	1
killed	1	2
let	2	1
me	1	1
noble	2	1
so	2	1
the	1	1
the	2	1
told	2	1
you	2	1
was	1	1
was	2	1
with	2	1

Basic Vector Space Model

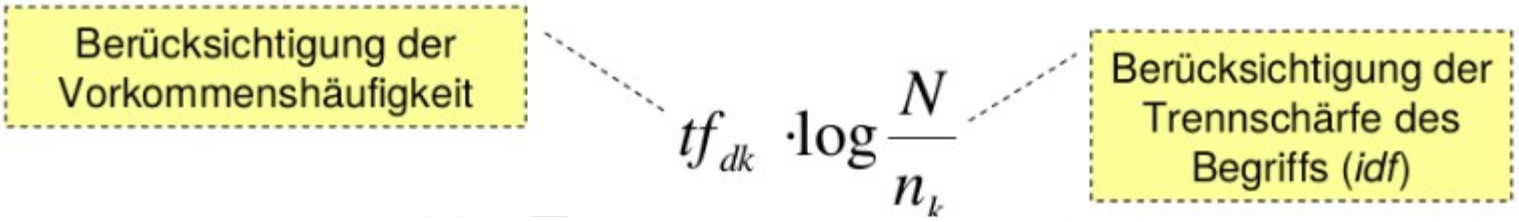


Vector Space Model



Funktiosweise der Informationsbeschaffung: Die tf-idf-Formel

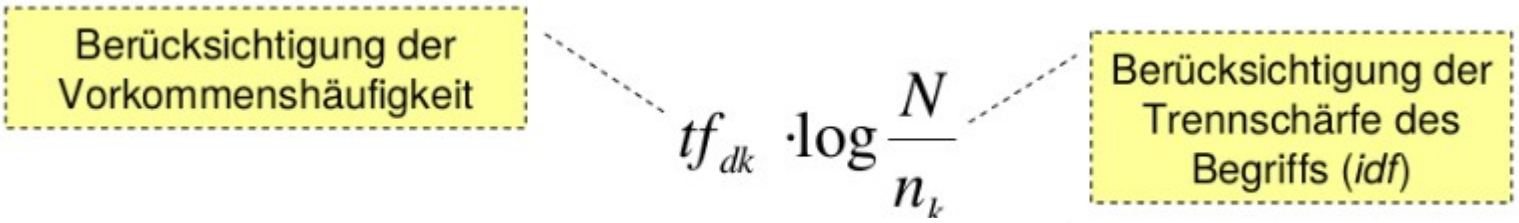
- Term Frequency- Inverse Dokument Frequency: die Seltenheit eines Begriffs innerhalb der Sammlung ist ein gutes Mass für die Bedeutung eines Begriffs



$$\text{tf} = \frac{\text{Häufigkeit des Suchbegriffs}}{\text{Häufigkeit des höchstfrequenten Begriffs}}$$

$$\text{idf} = \log \frac{\text{Anzahl der Dokumente}}{\text{Anzahl der Dokumente, die den Suchbegriff enthalten}}$$

Funktiosweise der Informationsbeschaffung: Die tf-idf-Formel



- **Beispiel:**
- Dokument mit Wörter mit deren Frequency:
- Kent = 3, Ohio = 2, University = 1
- Sammlung von 10000 Dokumente und Dokumentfrequenze:
- Kent = 50; Ohio = 1300; University = 250
- **THEN**
- Kent: tf = 3/3; idf = log(10000/50) tf-idf=5.3
- Ohio: tf=2/3; idf=log(10000/1300); tf-idf=1.3
- University: tf = 1/3; idf=log(10000/250); tf-idf=1.2

Evaluierung von IR-Systemen: Recall und Precision

- Ist das Objekt **relevant** oder **nicht relevant?**.
- **Wie vollständing ist das Ergebnis? → RECALL**
  - Anteil der gefundenen relevanten Dokumente an der Gesamtzahl der relevanten Dokumenten
  - 50 relevante gefunden; 100 relevante insgesamt → **Recall** = 50%
- **Wie genau ist das Ergebnis? Enthält es nur relevante Einträge? → PRECISION**
  - Anteil der relevante Dokumente im Ergebnis
  - 50 relevante gefunden; 75 insgesamt im Ergebnis → **Precision** = 67%

Evaluierung von IR-Systemen: Recall und Precision

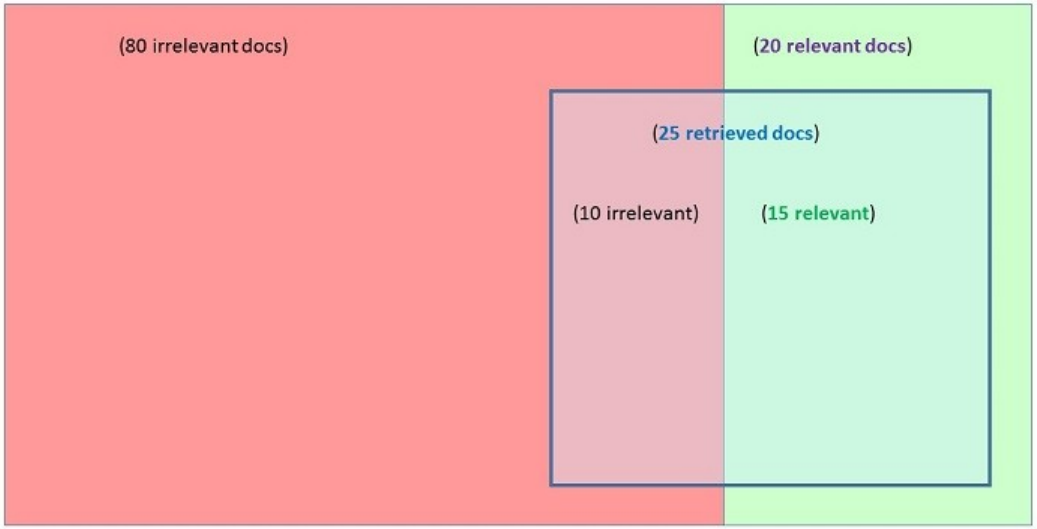
- Recall R: Anzahl der relevanten Dokumente im Ergebnis / Gesamtzahl der relevanten Dokumente
- Precision P: Anzahl der relevanten Dokumente im Ergebnis / Gesamtzahl der Dokumente im Ergebnis

	relevant	nicht relevant
im Ergebnis	tp	fp
nicht im Ergebnis	fn	tn

- Recall R = tp / (tp + fn)
- Precision P = tp / (tp + fp)

Evaluierung von IR-Systemen: Recall und Precision

- Precision: Anteil der abgerufenen Dokumente, die für die Informationsbedürfnisse des Benutzers relevant sind (correct/all).
- Recall: Anteil relevanter Dokumente in der Sammlung, die abgerufen werden (correct/should have been found)



Prec = retrieved/alle = 15/25=0.6
Rec = retrieved/relevante = 15/20 =0.75

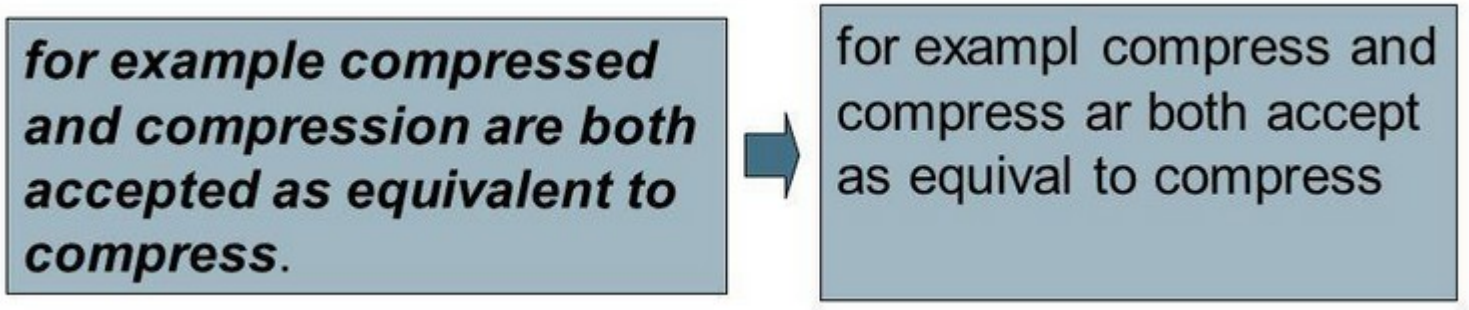
Parsen der Dokumente

- Daten sind häufig unstrukturiert (Format: pdf, doc, xls, html, usw.)
- Sprache des Dokuments
- Konvertierung und Bereinigung der Daten ist Grundlage für ein qualitativ hochwertiges IR-System
- Daten können zusätzlich veredelt werden
  - Anreicherung durch Begriffe aus Fachvokabularen
  - Synonyme, untergeordnete Begriffe, Begriffe in anderen Sprachen

  - Methode:
  - 1. End of Sentence (EOS) Detection
  - 2. Stemming
  - 3. Tokenization
  - 4. Part-of-Speech Tagging
  - 6. Chunking
  - 7. Extraction

Stemming

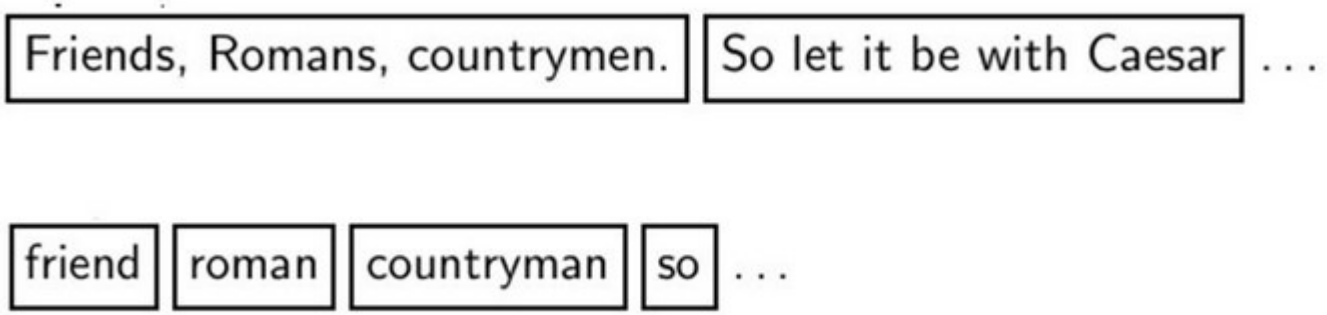
- Die Wörter werden auf ihre grammatikalische Grundform zurückgeführt
- Substantive auf den Nominativ Singular: Häuser → Haus
- Verben auf den Infinitiv: erprobte → erproben
- Stammformreduction: computer, computation, computerization → comput



Stemming: Sprachenprobleme

- In Englisch wenige Regeln decken die meisten Fälle
- In Deutsch → Stammänderungen für viele Wörter
- Umlauts → Haus – Häuser
- neue Präfixe → laufen – gelaufen
- Trennbare Verben → mitbringen – er brachte den Brief mit; überbringen – er überbrachte den Brief
- Schwein-kram, Schwein-s-haxe, Schwein-e-braten
- Konjugation: laufen - lief – gelaufen
- Deklination: das Haus, des Hauses, die Häuser
- Derivationsformen: proben, Erprobung, Probe

Tokenization



- Jeder Token ist ein Kandidat für einen Indexeintrag
- Welche Tokens können ausgelassen werden?
  → Terme die wenig Semantik tragen (and, or, if, usw.)

Tokenization: Sprachenprobleme

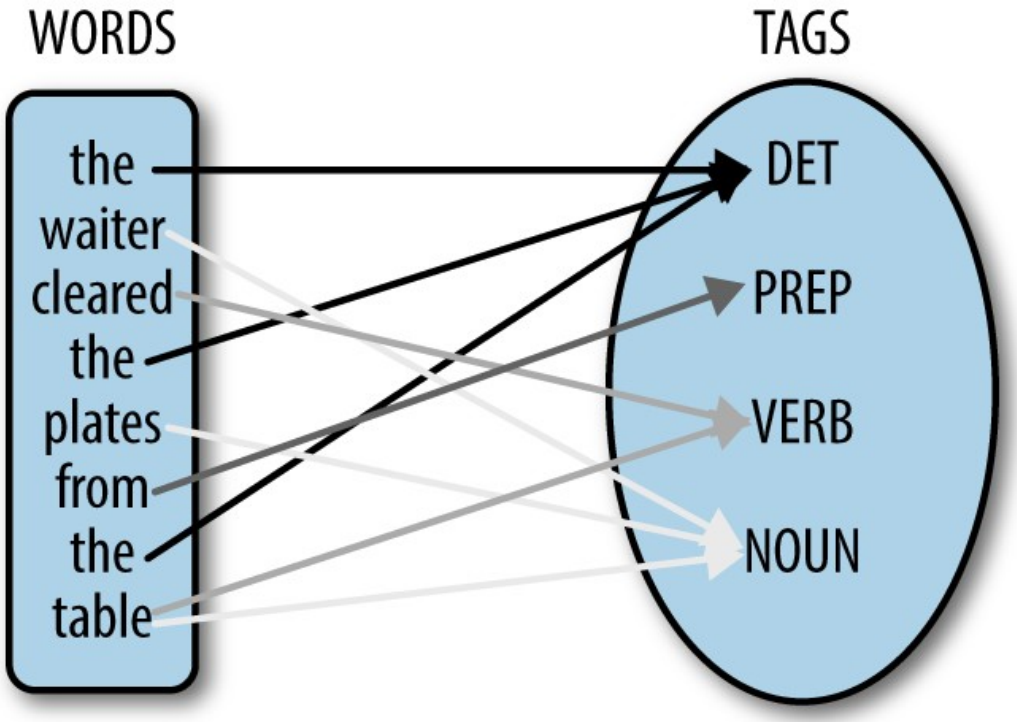
- New York → One Token or Two?
- Jammu & Kashmir → One Token or Three?
- Huh, Hmmm, Uh → ??
- India's Economy → India? Indias? India's?
- Won't, isn't → Will not? Is not? Isn't?
- Mother-in-law → Mother in Law?
- Ph.D. → PhD? Ph D ? Ph.D.?

Tokenization: Sprachenprobleme

- French
  - ***L'ensemble*** → one token or two?
    - ***L*** ? ***L'*** ? ***Le*** ?
    - Want ***l'ensemble*** to match with ***un ensemble***

- German noun compounds are not segmented
  - ***Lebensversicherungsgesellschaftsangestellter***
  - 'life insurance company employee'
  - German information retrieval needs **compound splitter**

Part of Speech Tagging



Chunking

Ungrouped	Randomly grouped	Rationally grouped
Bread	Bread	3 Dairy: Milk, Cream, Butter
Parsley	Parsley	
Milk	Milk	
Pasta	Pasta	4 Produce: Parsley, Carrots, Tomatoes, Beets
Carrots	Carrots	
Tomatoes	Tomatoes	
Cream	Cream	
Beets	Beets	3 Starches: Bread, Pasta, Flour
Butter	Butter	
Flour	Flour	

Thesauri

- Behandlung von Synonymen und Homonymen
- Manuell erstellt / gepflegte Äquivalenzklassen (**car = automobile, color = colour**)
- 2 Möglichkeiten:
  - Indexierung der Äquivalenzklassen: Dokument enthält **automobile**, **car** wird indexiert
  - Anfragenerweiterung: Anfrage enthält **automobile**, suche ebenso nach **car**

Zipf’sche Gesetzt

- Aussage des Zipfschen Gesetzes: Wenn die Wörter eines Textes nach ihrer Häufigkeit geordnet werden, ist die Wahrscheinlichkeit ihres Auftretens umgekehrt proportional zur Position innerhalb der Rangfolge:

$$p(n) \sim \frac{1}{n}.$$

Rang r einer Wortform in der Liste multipliziert mit seiner Häufigkeit n ist in etwa konstant.

rxn~k

Wortform	Häufigkeit n	Rang r	r ×n
sich	1.680.106	10	16.801.060
immer	197.502	100	19.750.200
Mio	36.116	500	18.059.500
Medien	19.041	1.000	19.041.000
Miete	3.755	5.000	18.775.000
vorläufige	1.664	10.000	16.640.000

Rang	Wortform	Anzahl
1	der	7377897
2	die	7036092
3	und	4813169
4	in	3768565
5	den	2717150
6	von	2250642
7	zu	1992268
8	das	1983589
9	mit	1878243
10	sich	1680106
11	des	1646885
12	auf	1640124
13	für	1638774
14	ist	1633510
15	im	1626923