Wissensrepräsentation
Vorlesung

Sommersemester 2008

9. Sitzung

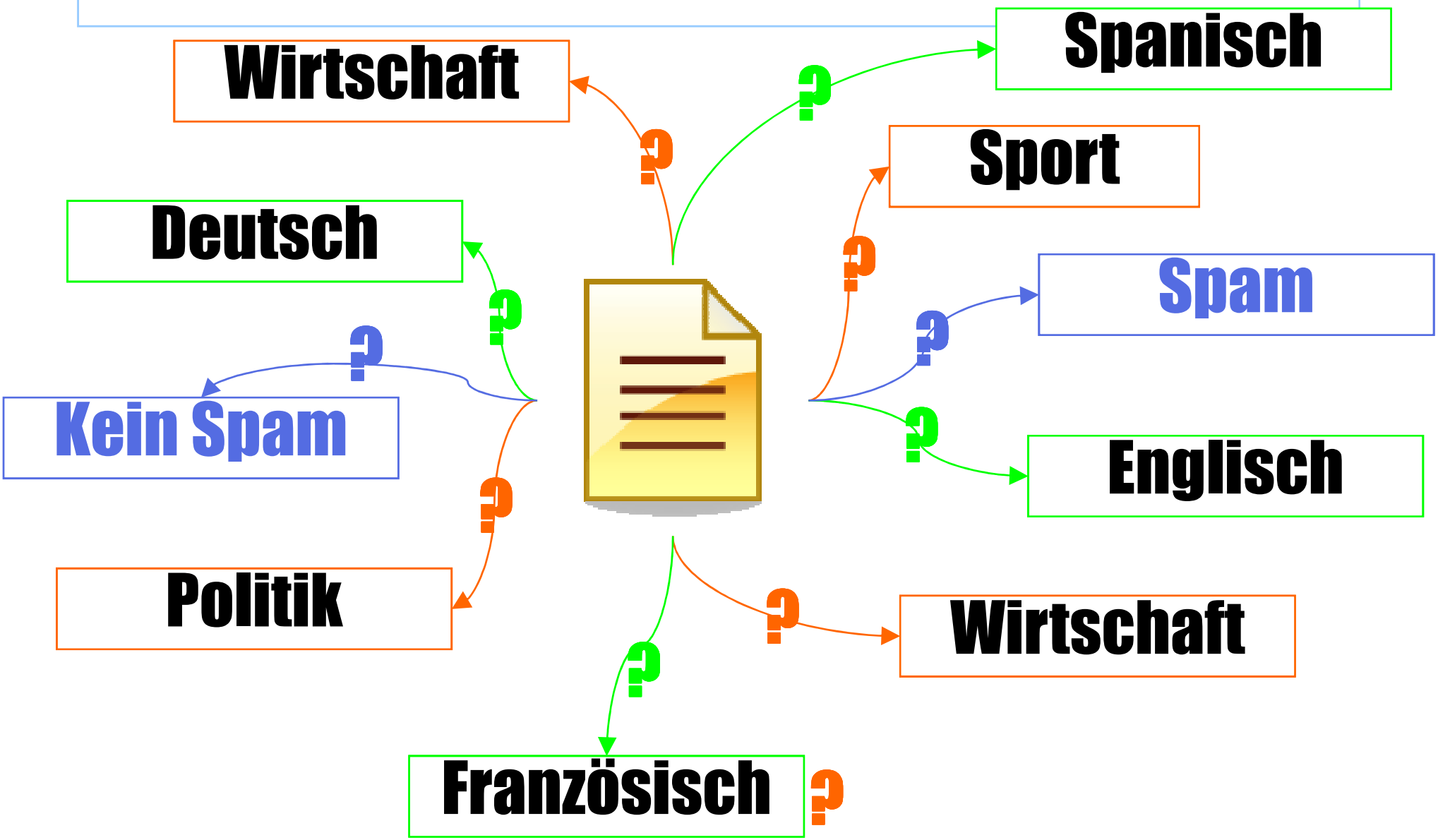
Dozent

Nino Simunic M.A.

Computerlinguistik, Campus DU

- **Statistische Verfahren der KI (II)**
  - **Klassifizieren von Dokumenten**
  - **Informationsbeschaffung**

Klassifizieren von Text



Erkennen der Sprache eines Texts

- Eine Möglichkeit:
  - Vollständiges Lexikon aller Sprachen der Welt
  - Regelbasiertes Vorgehen
  - Vermutlich sehr gute Ergebnisse
  - Vermutlich auch immenses Lexikon, um Vorhandensein aller Wörter zu garantieren

Schwierigkeiten beim Klassifizieren von Text

William B. **Cavnar**, John M. **Trenkle. 1994**:

- *One **difficulty in handling** some classes of **documents** is the presence of different kinds of **textual errors**, such as **spelling and grammatical errors in email**, and **character recognition errors** in documents that come through OCR.*
- ***Text categorization** must work reliably on all input, and thus **must tolerate** some level of **these kinds of problems**.*

  Regelbasierte bzw. rein symbolische Verfahren hätten hiermit Schwierigkeiten

Klassifikation mit statistischem n-Gramm Sprachmodell

William B. **Cavnar**, John M. **Trenkle. 1994.**, Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval. http://citeseer.ist.psu.edu/68861.html

- Statistisches, überwachtes Lernverfahren zur Klassifizierung von Dokumenten auf Basis von n-Gramm Statistiken bzw. solcher Profile
- Prinzipiell für die Klassifizierung von heterogenen Kategorie-Typen generisch einsetzbar
  - Sprache, Thema, Spam-NoSpam, …
  - Fokus hier: Sprache

N-Gramm (allgemein)

- Sequenz von (in der Regel) adjazenten Folgen
  - Je nach sprachtechnologischer Anwendung z.B. Buchstaben, Silben, Wörter, Sätze, …
- **n** bezeichnet die Länge einer Folge von »Geschriebenem«
- Je nach Anwendung unterschiedliche textuelle Einheiten (Zeichen, Wörter, …)
  - **Wort n-Gramme für »Eine tolle Sache ist das«**
    - **Unigramm (n=1): `{Eine, tolle, Sache, ist, das}`**
    - **Bigramm (n=2): `{Eine tolle, tolle Sache, Sache ist, ist das}`**

n-Gramme in Cavnar/Trenkles System

- Buchstaben / Zeichen-folgen verschiedener Länge (n=1 .. 5)
- Die Annahme bzw. das Sprachmodel

  Eine Sprache lässt sich durch ihre häufigsten Buchstaben bzw. Zeichen-Folgen beschreiben (unabhängig vom Texttyp):

  > The **top 300** or **so N-grams** are almost always **highly correlated to the language**.
  >
  > That is, a long **English passage about compilers** and a long **English passage about poetry** would tend to have a great **many N-grams in common in the top 300** entries of their respective profiles.
  >
  > On the other hand, a long passage in French on almost any topic would have a very different distribution of the first 300 N-grams.

  William B. **Cavnar**, John M. **Trenkle. 1994**:

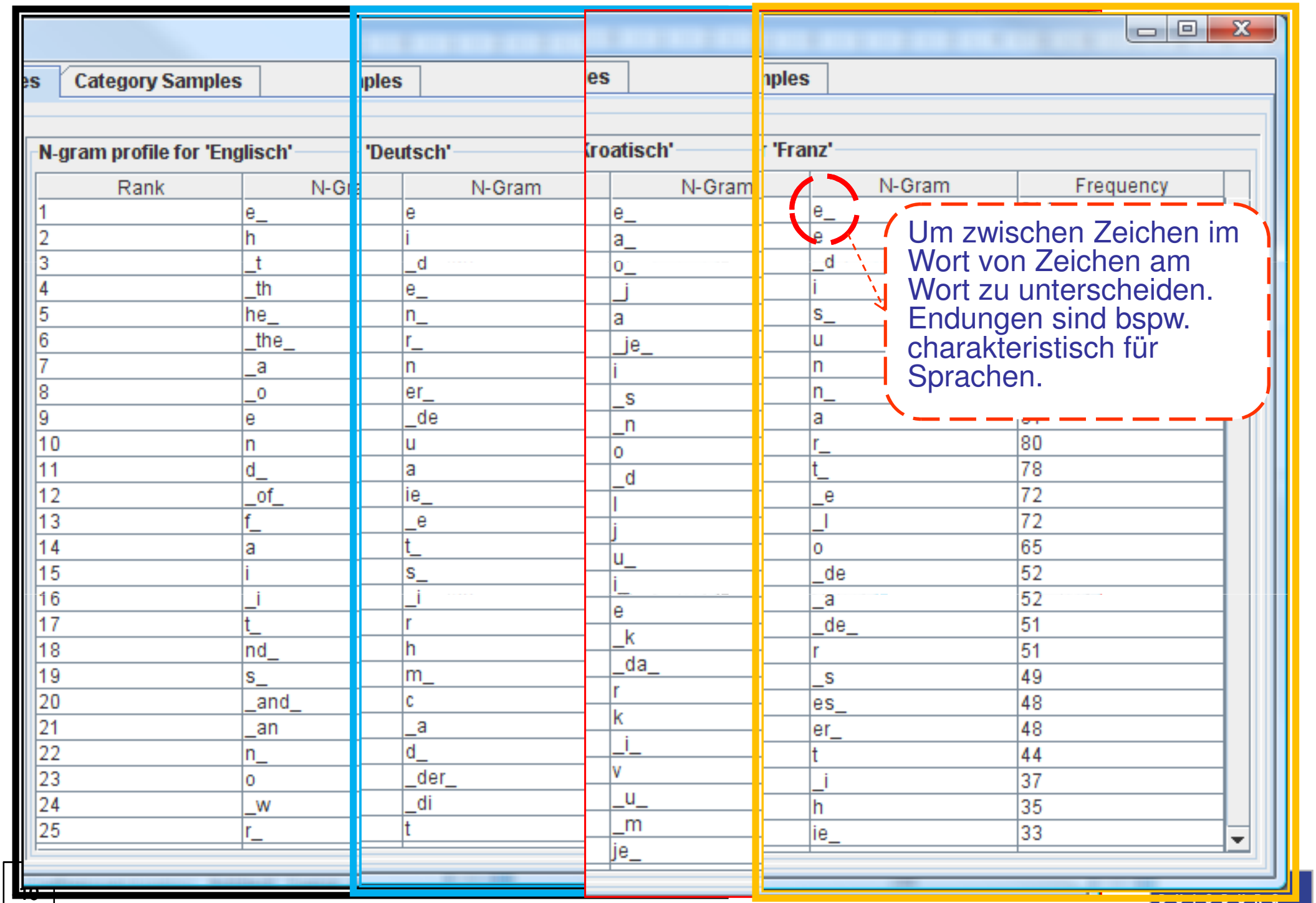
Beispiel für die häufigsten **Wort-Unigramme** in vier Sprachen:

Sortiert nach absteigender Häuifgkeit. Basis sind Textsammlungen in den verschiedenen Sprachen, die den allgemeinen Sprachgebrauch wiederspiegeln.

Warum nicht bei Cavnar/Trenkle? Wörter sind seltener als nur Teile dessen. Gefahr also, dass die trainierten Profile (viele) Wörter nicht enhalten und die Klassifizierung verfälschen.

Eng	Fre	Ger	Ita
the	de	der	di
and	la	die	e
to	le	und	il
of		den	che
a	et	in	la
in	des	von	a
was	les	.	in
his	du	zu	per
that	"	dem	del
I	en	,	un
he	un	fr	
as	que	mit	non
had	a	das	i
with	qui	des	si
it	dans	ist	le

Zeichen-N-Gramme mit unterschiedlicher Länge für vier Sprachen



Cavnar/Trenkle: »The system is based on **calculating** and comparing **profiles of N-gram frequencies**«

Zwei Phasen: Training, Anwendung

(1) Für jede Kategorie: Berechne (n-Gramm) Profil aus Trainingsdaten, welche die jeweilige Kategorien beschreibt. → **Kategorie-Profile**

(2) Erstelle Profil für eine Dokument, das kategorisiert werden soll und bestimme die Kategorie.

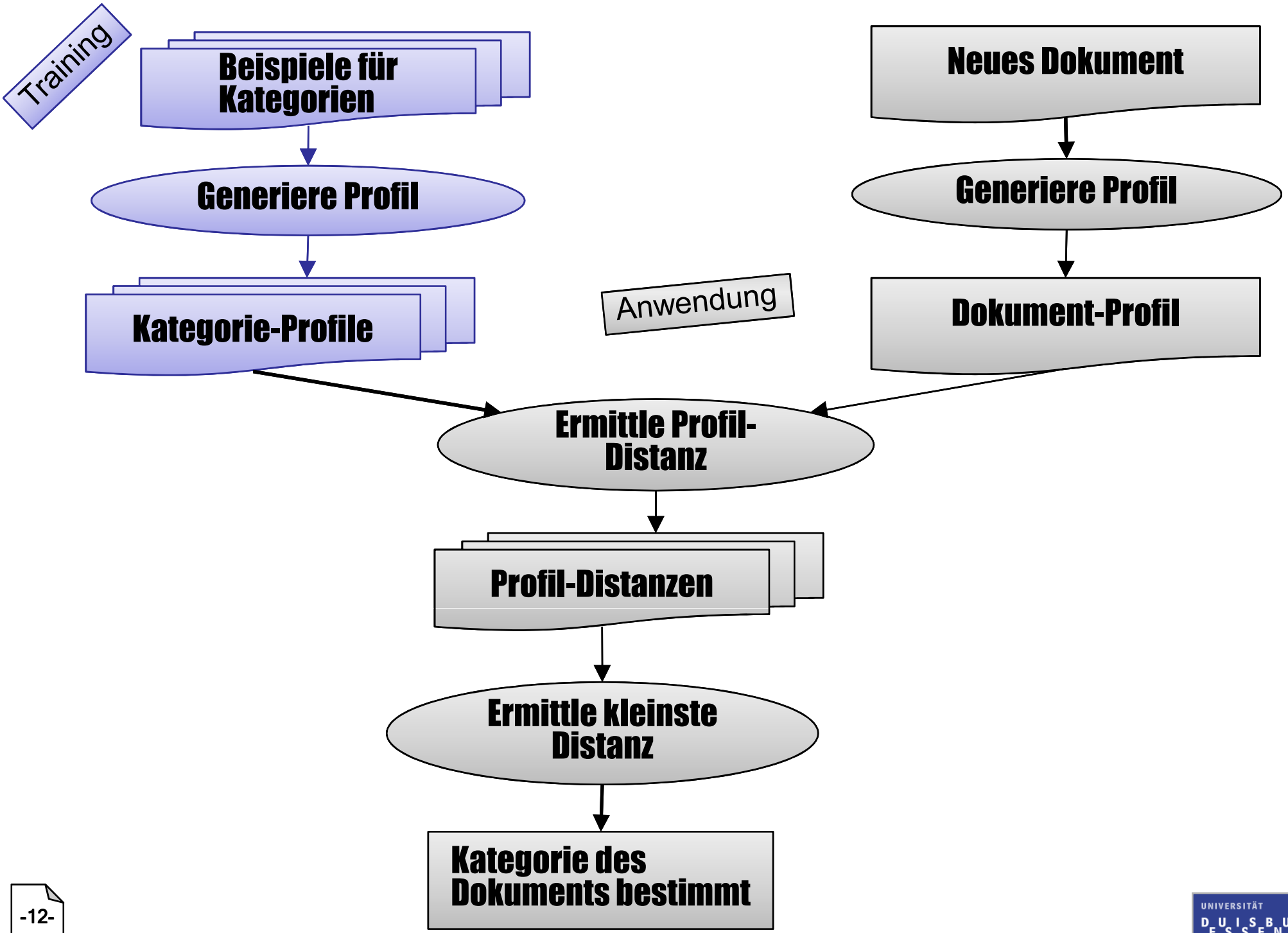
→ **Dokument-Profil**

(a) System errechnet eine **Distanz** zischen dem Dokument-Profil und jedem der Kategorie-Profile

(b) System wählt diejenige **Kategorie**, welche die **kleinste Distanz zum Dokument-Profil hat**.

- Ökonomisch: 10K bytes für jede Kategorie beim Training.

Ablauf am Beispiel: Dokument → Sprache

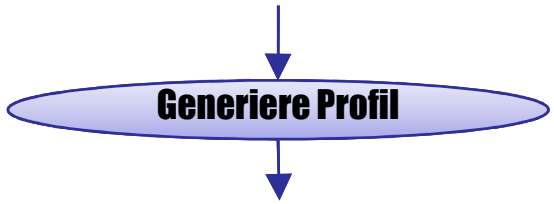


Verarbeitungsschritte beim Erstellen von Profilen

- Text wird tokenisiert (laufende Wörter werden strukturiert hinterlegt)
  - »*Das Ding kostet mich 1234$, toll!*«
  - → {*Das*, Ding, *kostet*, *mich*, *1234$*, , , *toll*, *!*}
- Text/Token werden normalisiert, Sonderzeichen, Zahlen entfernt
  - → {Das, Ding, kostet, mich, toll}
- Token werden in mögliche Zeichen-N-Gramme zerlegt (n=1 .. 5)
- Zählen, wie häufig jedes eindeutige N-Gramm in einer Sprache vorkommt.

→ Ranking der N-Gramme.

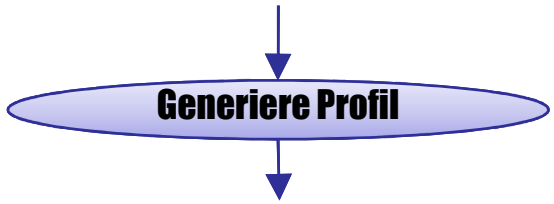
(Sehr) Simples Beispiel



- Eine N-Gramm-Statistik-Profil für die Phantasiesprache *Affistanisch* wird auf Basis eines Trainingstexts mit Inhalt »Banane« trainiert (**N=1..5**)

UniGramme (n=1):	**{_B,a,n,a,n,e_}**
BiGramme (n=2):	**{_Ba, an, na, an, ne_}**
TriGramm (n=3):	**{_Ban, ana, nan, ane_}**
TetraGramme (n=4):	**{_Bana, anan, nane_}**
PentaGramme (n=5):	**{_Banan, anane_}**

»Banane«: Sortierung der N-Gramme nach (absteigender) Häufigkeit



UniGramme (n=1):	{_B,a,n,a,n,e_}
BiGramme (n=2):	{_Ba, an, na, an, ne_}
TriGramm (n=3):	{_Ban, ana, nan, ane_}
TetraGramme (n=4):	{_Bana, anan, nane_}
PentaGramme (n=5):	{_Banan, anane_}

Wird für jedes Kategorie-Profil einmalig erstellt und gespeichert.

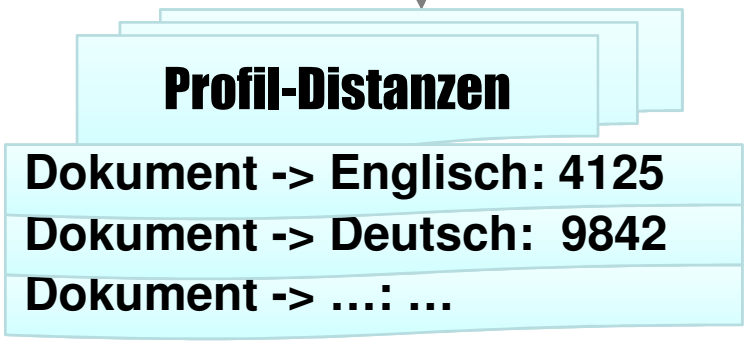
Ein zu klassifizierendes Dokument wird analog dazu verarbeitet, und ein Profil erstellt.

n	(2)
a	(2)
an	(2)
ane_	(1)
anan	(1)
anane_	(1)
ana	(1)
e_	(1)
_B	(1)
_Banan	(1)
…	

Zurück zum eigentlich System und der Anwendung nach erfolgtem Lernen

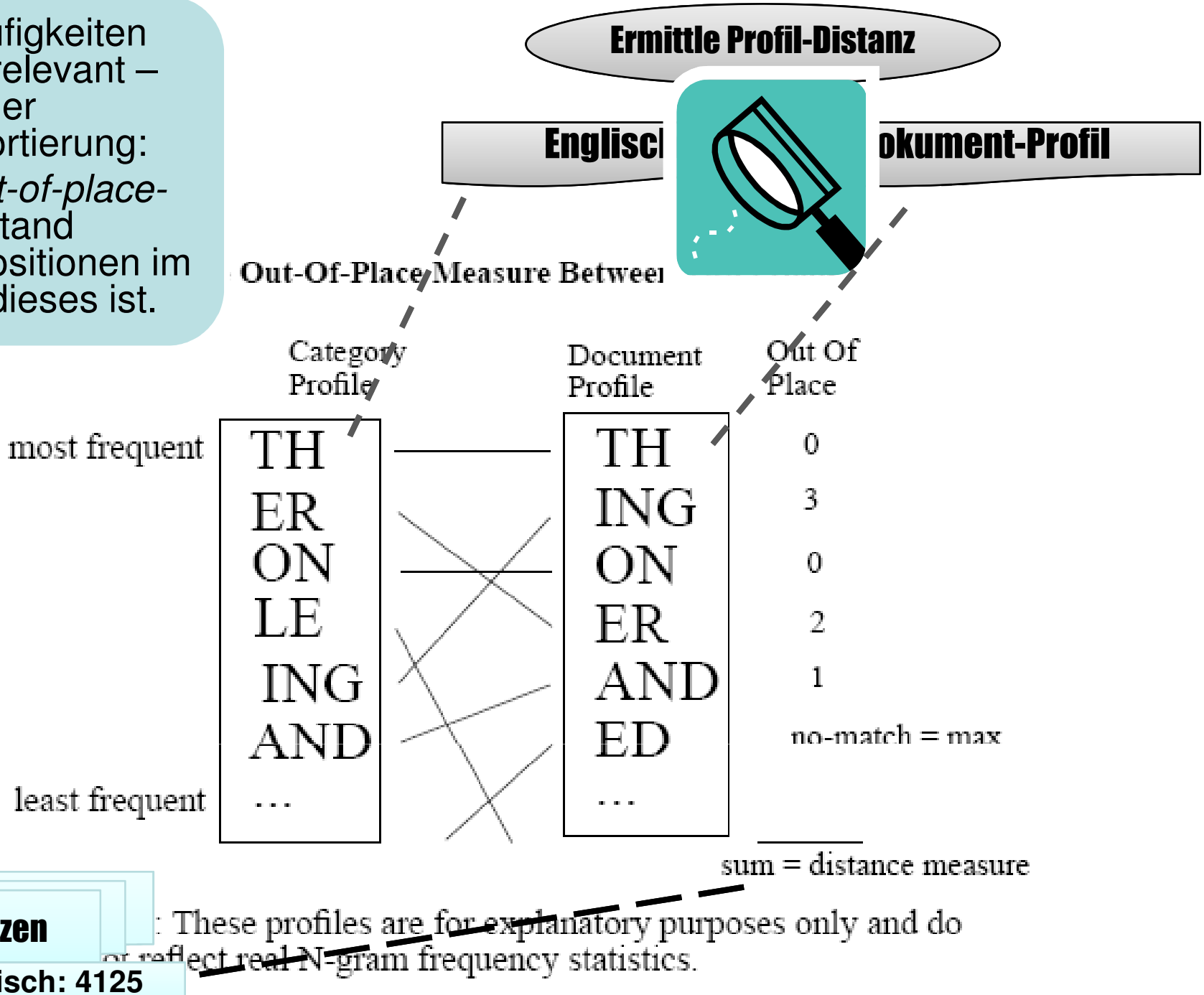
Dokument-Profil

Ermittle Profil-Distanz



Wichtig: Die Häufigkeiten sind *nicht mehr* relevant – sie dienten nur der absteigenden Sortierung:

Basis für das *out-of-place*-Maß, da der Abstand zwischen den Positionen im Ranking genau dieses ist.

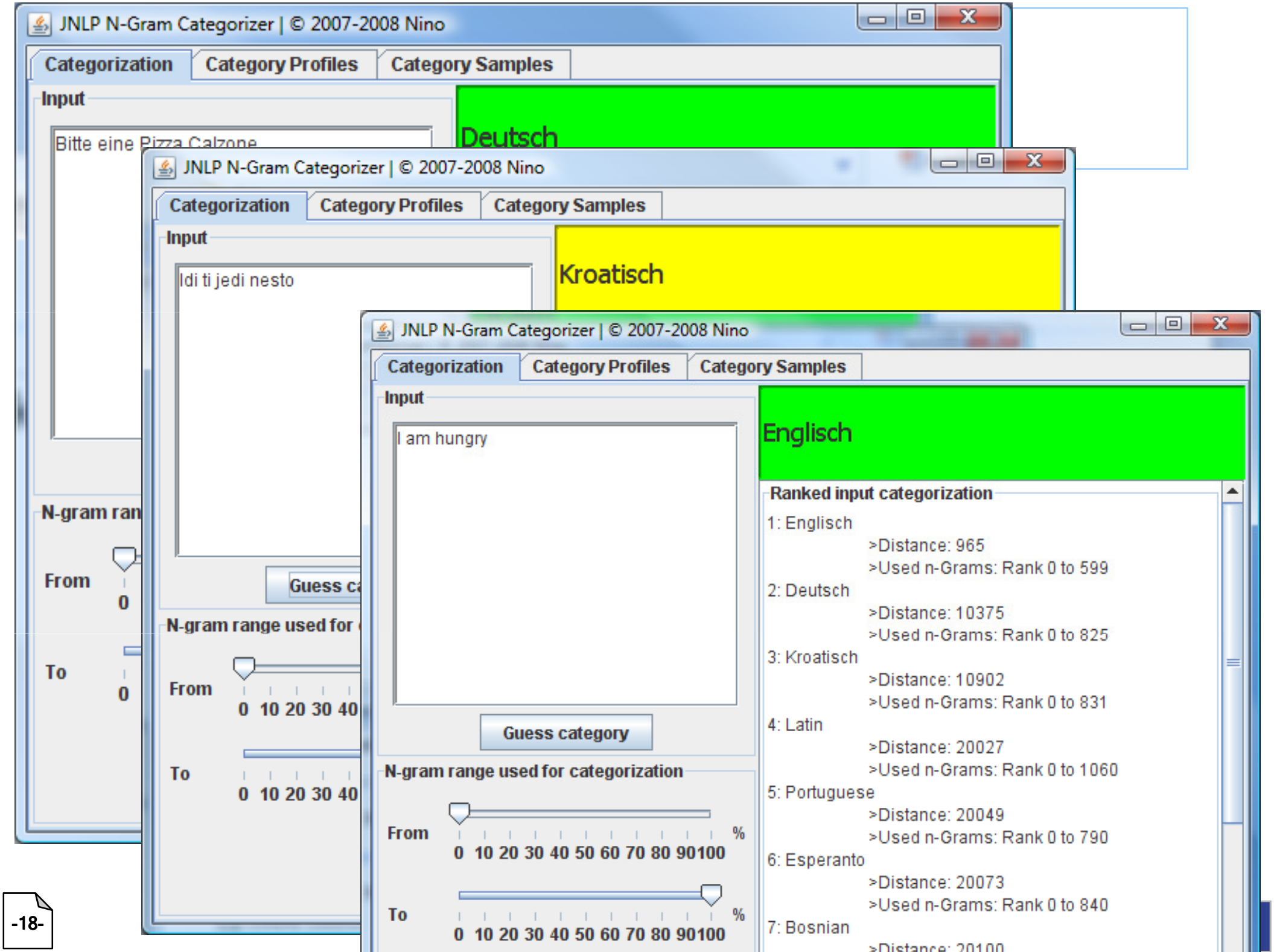


Profil-Distanzen

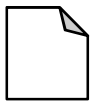
Dokument -> Englisch: 4125

Dokument -> ...: ...

Cavnar, Trenkle: 1994


JNLP N-Gram Categorizer | © 2007-2008 Nino
Categorization
Category Profiles
Category Samples
Input
Bitte eine Pizza Calzone
Deutsch
N-gram ran
From
0
To
0
JNLP N-Gram Categorizer | © 2007-2008 Nino
Categorization
Category Profiles
Category Samples
Input
Idi ti jedi nesto
Kroatisch
Guess ca
N-gram range used for
From
0 10 20 30 40
To
0 10 20 30 40
JNLP N-Gram Categorizer | © 2007-2008 Nino
Categorization
Category Profiles
Category Samples
Input
I am hungry
Guess category
N-gram range used for categorization
From
0 10 20 30 40 50 60 70 80 90100
%
To
0 10 20 30 40 50 60 70 80 90100
%
Englisch
Ranked input categorization
1: Englisch
>Distance: 965
>Used n-Grams: Rank 0 to 599
2: Deutsch
>Distance: 10375
>Used n-Grams: Rank 0 to 825
3: Kroatisch
>Distance: 10902
>Used n-Grams: Rank 0 to 831
4: Latin
>Distance: 20027
>Used n-Grams: Rank 0 to 1060
5: Portuguese
>Distance: 20049
>Used n-Grams: Rank 0 to 790
6: Esperanto
>Distance: 20073
>Used n-Grams: Rank 0 to 840
7: Bosnian
>Distance: 20100

- Statistische Verfahren der KI (II)
  - Klassifizieren von Dokumenten
  - **Informationsbeschaffung**



Statistische Verfahren im Information Retrieval: Ad hoc Retrieval

- **Information Retrieval, Informationsbeschaffung**
  - Ziel: Dem Benutzer inhaltlich relevante Informationen (schnell) aus großen Datenmengen zu beschaffen.
  - Grundlage vieler ad hoc IR-Systeme: Bag of words, Vektoren
- Hier/Wir: Ad hoc IR mit binären und »echten« Merkmalen in mehrdimensionalen Vektor-Räumen

Bedeutung von Dokumente, Bag of words

- Die Bedeutung eines Dokuments basiert ausschließlich auf den enthaltenen Wörtern
  - »[E]xtreme **interpretation of compositional semantics**« **[JURMAR:2000]**
  - Gesamtbedeutung von *Cesar sold Marcus* ergibt sich aus der Komposition der Einzelbedeutungen
- Wie »extreme« sind solche Systeme?
  - Zwischen *Cesar sold Marcus* und *Marcus sold Cesar* wird nicht differenziert

»Bag of words« Verfahren

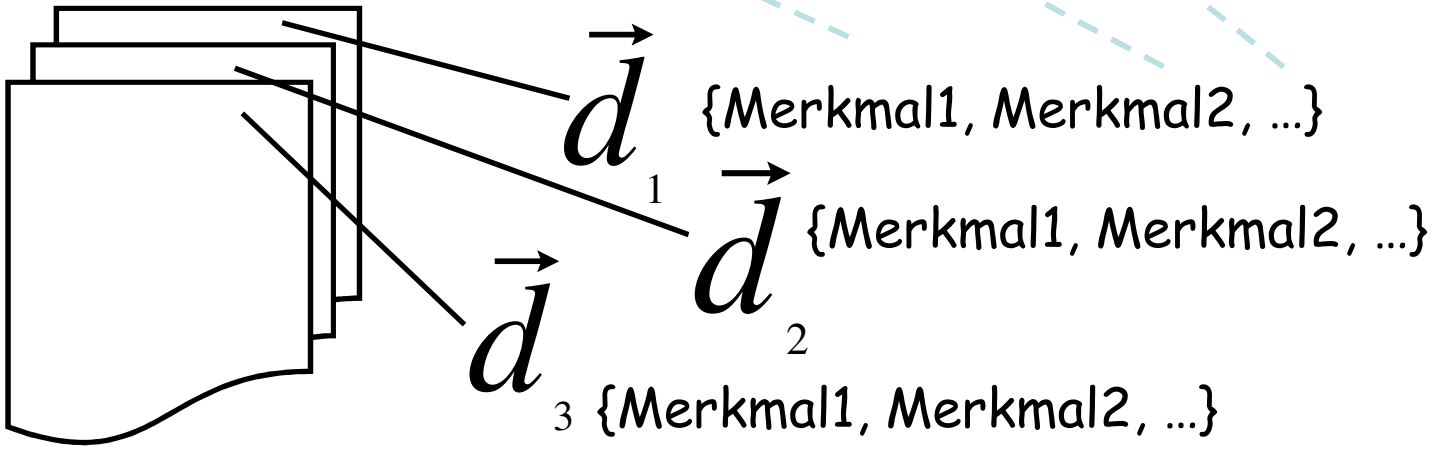
Syntaktische (, semantische, pragmatische, …) Informationen fehlen vollständig.

Vorgehen

- Sammlung von Dokumenten, die durchsucht werden können
- »**Wichtige**« **Begriffe** werden aus den Dokumenten extrahiert (und gezählt)
  - **Inhaltswörter**
  - Filterung : Stoppwörter bzw. Listen mit solchen
  - Oder: Weitere (statistische) Verfahren zur Extraktion von Schlüsselwörtern bzw. zur Gewichtung von Termen
    - Populär: tf*idf

Vokabular, Datenstrukturen

- Datenstruktur mit allen »wichtigen« Begriffen der gesamten Sammlung
  { Hund, Katze, Maus, … }
- Jedes Dokument mit eigener Datenstruktur (**Vektor**) und **Merkmalen** als Werte



Vektoren mit Merkmalen

Repräsentation eines Dokuments *j*

Vektor für ein Dokument *j* mit den Merkmalen 1-N

$$\vec{d}_j = (t_{1,j}, t_{2,j}, t_{3,j}, \ldots, t_{N,j})$$

Vektor für Query *k* mit den Merkmalen 1-N

Kann neben Anfrage auch ein Dokument sein.

$$\vec{q}_k = (t_{1,k}, t_{2,k}, t_{3,k}, \ldots, t_{N,k})$$

Repräsentation einer Query *q*

d für **d***ocument*, **q** für **q***uery*, **t** für **t***erm*.

Binäre/Boolesche Merkmale am einfachen Beispiel

Binäre t-Merkmale: {0,1}

1: Term tritt in Dokument/Query auf

0: Term tritt ***nicht*** in Dokument/Query auf

Indizes der Elemente im Sammlungsvektor müssen denen der Dokumentenvektoren entsprechen.

Mini-Textsammlung mit Dokumenten x,y

d_x: Krise … Öl … Nahen … Osten

d_y: …Krise …Öl …Irak

Die N Begriffe d. Sammlung:
Krise, Öl, Nahen, Osten, Irak

$$\vec{d}_x = (1,1,1,1,0)$$

Krise, Öl, Nahen, Osten, Irak

$$\vec{d}_y = (1,1,0,0,1)$$

Beispiel-Anfrage: »Krieg Öl Osten«

- Wie Dokuemten:Vektor mit binären Merkmalen

Beispiel 1: Query als Vektor mit binären Merkmalen

q_k **Krieg Öl Osten**

$$q_k = (0,1,0,1,0)$$

Krise, Öl, Nahen, Osten, Irak

Die N Begriffe d. Sammlung:
Krise, Öl, Nahen, Osten, Irak

Ähnlichkeitsmessung (2)

- Bestimmung der Relevanz zwischen Anfrage und Dokumenten in der Sammlung?
  - Anzahl der Merkmale, die die Vektoren $\mathbf{q_k}$ und jeweils die Vektoren $\mathbf{d_x}$, $\mathbf{d_y}$ gemeinsam haben.
- Berechnung des Ähnlichkeitsmaß zwischen Query und Dokument (Skalarprodukt, engl. *dot product*):

$$sim(\vec{q}_k, \vec{d}_x) = \sum_{i=1}^{N} t_{i,k} \times t_{i,x}$$

- Beispiel: Wie ähnlich sind die $\mathbf{d}$-Vektoren und $\mathbf{q}$?

Beispiel 1: Ad hoc Retrieval und Ähnlichkeitsmessung (3)

Krise, Öl, Nahen, Osten, Irak

Krise, Öl, Nahen, Osten, Irak

$$\vec{q}_k = (0,1,0,1,0) \bullet \vec{d}_x = (1,1,1,1,0)$$

$$\vec{q}_k \cdot \vec{d}_x = (0)(1) + (1)(1) + (0)(1) + (1)(1) + (0)(0)$$

$$= 0 + 1 + 0 + 1 + 0 = \underline{\underline{2}}$$

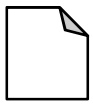
"Krise Öl Nahen Osten"
ist "Krieg Öl Osten"
demnach am ähnlichsten

$$\vec{q}_k = (0,1,0,1,0) \bullet \vec{d}_y = (1,1,0,0,1)$$

$$\vec{q}_k \cdot \vec{d}_y = 0 + 1 + 0 + 0 + 0 = \underline{\underline{1}}$$

Probleme mit binärer Vektorisierung

- Mit dem binären Ansatz gehen wichtige Informationen verloren
  - Wir können Folgendes damit bspw. nicht ausdrücken, dass einige Wörter für ein Dokument wichtiger sind als andere
- Besserer Ansatz: Echte Werte (*real values*) nehmen, um graduell Wichtigkeit anzeigen zu können, bzw. um …
- … Termen **Gewichte** zu geben/sie zu **gewichten**



Generalsierung der Vektoren

$$\vec{d}_j = (w_{1,j}, w_{2,j}, w_{3,j}, \ldots, w_{n,j})$$

$$\vec{q}_k = (w_{1,k}, w_{2,k}, w_{3,k}, \ldots, w_{n,k})$$

$w_{i,j}$: Gewicht **i** in Dokument **j**

- Generalisierung auch als ***term-by-document-matrix*** vorstellbar:
  - Spalten repräsentieren Dokumente
  - Zeilen repräsentieren Gewichte

Vektoren im mehrdimens. Raum

- Vektoren entsprechen Punkten im mehrdimensionalen Raum
- Beispiel: **Drei**dimensionaler Raum
  - Merkmale sind jeweils (hier **drei**) Frequenzdaten zu ausgewählten Begriffen (*Heinz, Tomate, Nudeln)* aus einer Dokumentensammlung

$$\vec{d}_{doc\ 1} = (1,2,1)$$

$$\vec{d}_{doc\ 7} = (6,0,1)$$

$$\vec{d}_{doc\ 13} = (0,5,1)$$

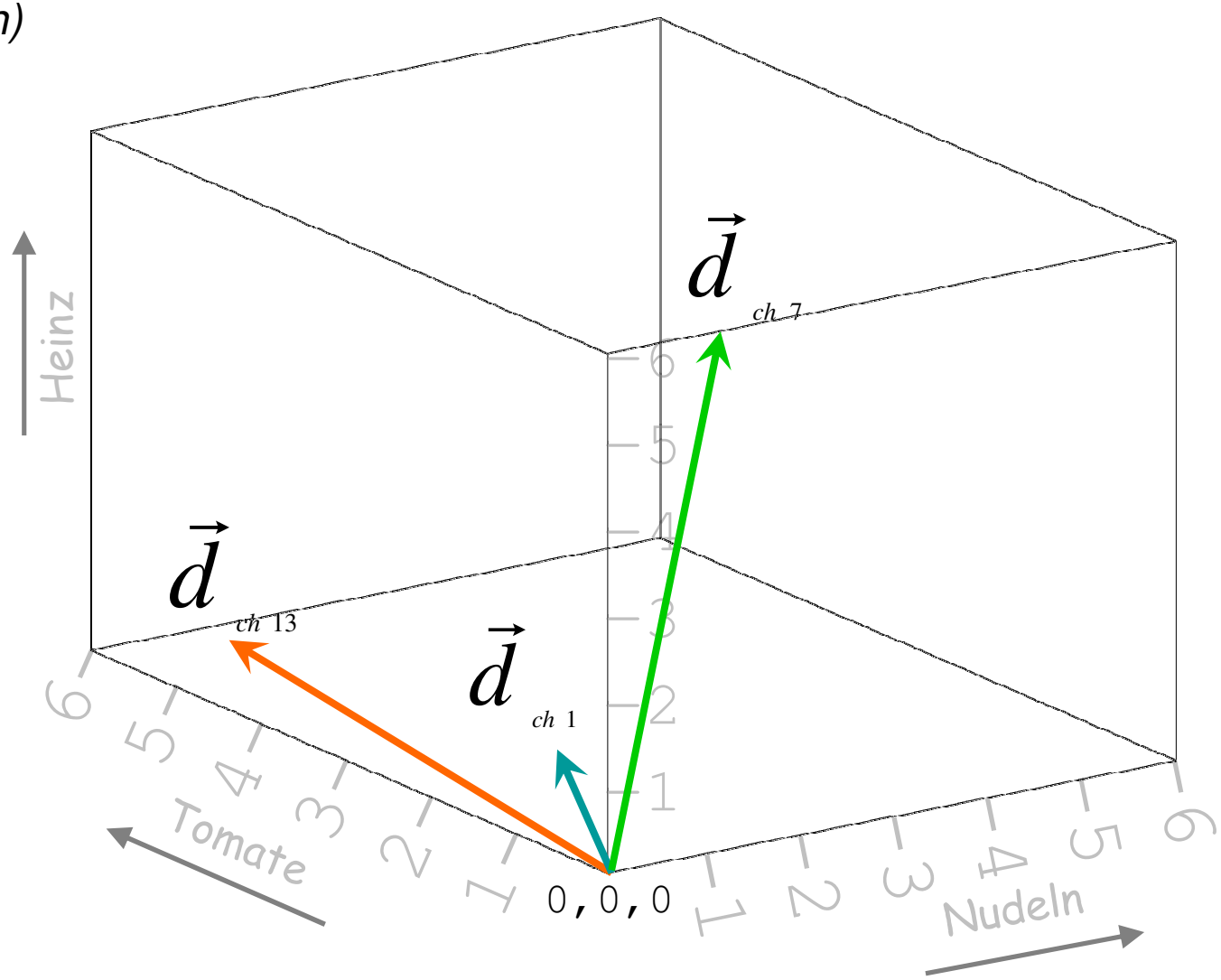
Beispiel: 3D -Vektorraum

(Heinz, Tomate, Nudeln)

$\vec{d}_{doc\ 1} = (1,2,1)$

$\vec{d}_{doc\ 7} = (6,0,1)$

$\vec{d}_{doc\ 13} = (0,5,1)$



Fehlende Normalisierung

- Problem, wenn Vektoren unterschiedlicher Längen vorliegen (letze Grafik), wir sie jedoch auf gleicher Grundlage vergleichen möchten. Z.B.:

$$\vec{q}_k = (1,2,1)$$

$$\vec{d}_x = (20\ ,42\ ,13\ )$$

$$\vec{d}_y = (37\ ,90\ ,21\ )$$

Die absoluten Werte sind die unterschiedliche Grundlage, die jedoch relational gleich sind.

Analogie

- Analogie: Der Preis eines Porsche ist für einen Konzernvorstand genauso so hoch wie für einen Obdachlosen.
- Das »Gefühl«, dass der Porsche für einen Bonzen »viel weniger« wert ist als für den Penner, kann man explizit machen, indem man den Preis in Relation zum jeweiligen Jahreseinkommen setzt und ihn so auf Prozentsatz-Basis **normalisiert**.

»Fehlende Normalisierung ist problematisch« am Beispiel:

(Irak, Iran, Krise)

$$\vec{q}_k = (1,2,1)$$

Der Anfrage-Vektor

$$\vec{d}_x = (20, 42, 13)$$

$$\vec{d}_y = (37, 90, 21)$$

Dokumentenvektoren der *collection*

$$\vec{q}_k \cdot \vec{d}_x = (1)(20) + (2)(42) + (1)(13)$$

$$= 20 + 84 + 13 = 117$$

Ähnlichkeitsmessung zwischen den Dok. und der Anfrage

$$\vec{q}_k \cdot \vec{d}_y = (1)(37) + (2)(90) + (1)(21)$$

$$= 37 + 180 + 21 = 238$$

Normalisierung

- Kurz: **Nicht die einzelnen Punkte im Raum interessieren** (absolute Werte, Länge der Vektoren), **sondern** ihre **Richtung**!
- **Wünschenswert**: **Einheitliche Vektorlängen** als Grundlage. Somit spielt die Länge keine Rolle mehr, sondern der Fokus liegt auf der Richtung.
- **Normalisierung** kann dabei **a priori** für die Dimensionen im Vektor geschehen, oder aber direkt in den Ähnlichkeitsvergleich zwischen ganzen Vektorem verwurstet werden.
- Zunächst die a priori Normalisierung der Vektor-Dimensionen.

Berechnung einheitlicher Vektorlängen

- Die **Konvertierung zur einheitlichen Länge** erfolgt, indem jede **Vektor-Dimension v_i durch die Gesamtlänge des Vektors $|\vec{v}|$ geteilt** wird:

$$\frac{v_i}{|\vec{v}|}$$

Die Länge eines Vektors errechnet sich wie folgt:

$$|\vec{v}| = \sqrt{\sum_{i=1}^{N} v_i^2}$$

Beispiel-Normalisierung

$$\vec{d}_{doc\ 1} = (1,2,1)$$

Normalisierungsbedürftiger Vektor

Berechnung der einheitlichen Vektor-Länge:

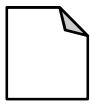
$$\left|\vec{v}\right| = \sqrt{\sum_{i=1}^{N} v_i^2}$$

$$\left|\vec{d}_{doc\ 1}\right| = \sqrt{1^2 + 2^2 + 1^2} = \sqrt{6} \approx 2{,}45$$

Normalisierter Vektor:

$$\vec{d}_{doc\ 1} = \left(\frac{1}{2{,}45}, \frac{2}{2{,}45}, \frac{1}{2{,}45}\right)$$

$$= (0.41, 0.82, 0.41)$$



Ähnlichkeitsmessung mit norm. Vektoren (1)

(Irak, Iran, Krise)

$$\vec{q}_k = (1,2,1)$$

$$\vec{d}_x = (20\ ,42\ ,13\ )$$

$$\vec{d}_y = (37\ ,90\ ,21\ )$$

Längenbestimmung:

$$\left|\vec{d}_x\right| = \sqrt{20^2 + 42^2 + 13^2} = \sqrt{2333} \approx 48{,}30$$

$$\left|\vec{d}_y\right| = \sqrt{37^2 + 90^2 + 21^2} = \sqrt{9910} \approx 99{,}55$$

Die normalisierten Vektoren:

$$\vec{d}_x = \left(\frac{20}{48.30}, \frac{42}{48.30}, \frac{13}{48.30}\right) = (0.41, 0.87, 0.27)$$

$$\vec{d}_y = \left(\frac{37}{99.55}, \frac{90}{99.55}, \frac{21}{99.55}\right) = (0.37, 0.90, 0.21)$$

Ähnlichkeitsmessung mit norm. Vektoren (2)

Basis: Alle Vektoren sind normalisiert worden

$$\vec{q}_k = (0.41, 0.82, 0.41)$$

$$\vec{d}_x = (0.41, 0.87, 0.27)$$

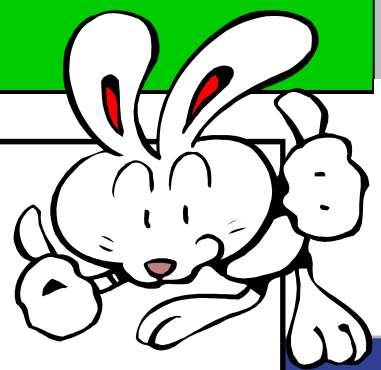
$$\vec{d}_y = (0.37, 0.90, 0.21)$$

Ähnlichkeitsmessung:

$$\vec{q}_k \cdot \vec{d}_x = (0.41)(0.41) + (0.82)(0.87) + (0.41)(0.27)$$
$$= 0.17 + 0.71 + 0.11 = 0.99$$

$$\vec{q}_k \cdot \vec{d}_y = (0.41)(0.37) + (0.82)(0.90) + (0.41)(0.21)$$
$$= 0.15 + 0.74 + 0.09 = 0.98$$

Die Dokumente ähneln sich nun sichtbar mehr, als bei der Berechnung mit absoluten Werten
(Vorher: `117 = q*d_x vs. 238 = q*d_y`**)**



Cosinus-Maß

- Unser heimlicher Begleiter: Das Cosinus-Maß
- Bei normalisierten Vektoren wird es wie schon bekannt berechnet:

$$\cos(\vec{q}, \vec{d}) = \vec{q} \cdot \vec{d}$$

Messung des Winkels zwischen Vektoren

→**Wenn Vektoren (Dokumente) identisch** sind (sie liegen räumlich übereinander), ist das **Cosinus-Maß gleich** `1.0`

→ **Wenn Vektoren »nichts« gemeinsam haben,** ist das **Cosinus-Maß gleich** `0.0` **(orthogonale Vektoren)**

- -1.0 entspräche Vektoren mit entgegengesetzter Richtung.

Referenzen

- William B. **Cavnar**, John M. **Trenkle. 1994**., Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval.
- D. **Jurafsky**; J.H. **Martin**. **2000**. Speech and Language Processing. Prentice-Hall.

Aufgabe

- Lesen (**Cavnar**, **Trenkle** **):**
  http://citeseer.ist.psu.edu/68861.html