Universität Potsdam
Institut für Informatik
Lehrstuhl Maschinelles Lernen

Statistische Sprachmodelle

Tobias Scheffer
Thomas Vanck

Statistische Sprachmodelle

- Welche Sätze sind Elemente einer Sprache (durch Grammatik einer Sprache beschrieben).
- Wie wahrscheinlich ist ein bestimmter Satz in einem Kontext
  - P(„ich pflücke Beeren“) ist vielleicht 0.0001,
  - P(„ich pflücke Bären“) ist vielleicht 10^{-25}.
- Statistische Sprachmodelle:
  - N-Gramme,
  - probabilistische kontextfreie Grammatiken.

Statistische Sprachmodelle: wozu?

- Spracherkennung:
  - Akustisches Modell + Sprachmodell.
- Handschrifterkennung:
  - Schreibmodell + Sprachmodell.
- Rechtschreibkorrektur:
  - Fehler- / Vertippmodell + Sprachmodell.
- Übersetzung:
  - Übersetzungsmodell + Sprachmodell der Zielsprache.
- PDA-Tastatureingabe :
  - Normalverteilte Stiftposition + Sprachmodell.
- …

Statistische Sprachmodelle: wozu?

- Bayes' Formel („DIE Formel"):

$$\arg\max_{w_1,\ldots,w_T} P(w_1,\ldots,w_T \mid signal)$$

$$= \arg\max_{w_1,\ldots,w_T} \underbrace{P(signal \mid w_1,\ldots,w_T)}_{\text{Akustikmodell (HMM)}} \underbrace{P(w_1,\ldots,w_T)}_{\text{Sprachmodell auf Wortebene}}$$

Spracherkennung:	Akustikmodell (HMM)	Sprachmodell auf Wortebene
Handschrifterkennung:	Schriftmodell	Sprachmodell auf Wortebene
Rechtschreibkorrektur:	Vertipp-/Fehlermodell	Sprachmodell auf Wortebene
Übersetzung:	Übersetzungsmodell	Sprachmodell auf Wortebene
PDA-Tastatureingabe:	Stiftpositionsmodell	Sprachmodell auf Buchstabenebene
T9	Buchst.-Tastenzuordnung	Sprachmodell auf Buchstabenebene

N-Gramm-Modelle

- P(„ich pflücke Beeren“) = P(„ich“) x P(„pflücke“ | „ich“) x P(„Beeren“ | „ich pflücke“).

- Markov-Annahme (N-1). Ordnung (bsp: 2. Ordnung):
  - P(„Wald“ | „ich pflücke Beeren im“) = P(„Wald“ | „Beeren im“).

- Modell speichert Wahrscheinlichkeiten für Wortfolgen der Länge maximal N.
- N=2: „Bigramm“ (richtig wäre eigentlich „Digramm“), N=3: „Trigramm“, N=4: „Tetragramm“.

N-Gramme

- Vorhersage des nächsten Wortes:
  - Bestimme $P(w_t \mid w_{t-1}, ..., w_1)$.
  - Die meisten Wortfolgen $w_1, ..., w_t$ tauchen in einem Korpus nur einmal auf,
  - wie können wir dafür Wahrscheinlichkeiten ermitteln?
- Markov-Annahme $(N\text{-}1)$-ter Ordnung:
  - Nur die letzten n-1 Wörter sind entscheidend für das nächste (n-te) Wort.
  - $P(w_{t+N} \mid w_{t+N-1}, ..., w_1) = P(w_{t+N} \mid w_{t+N-1}, ..., w_t)$

Markov-Annahme $(N\text{-}1)$-ter Ordnung

- Wahrscheinlichkeit für Wort $t{+}N$ ergibt sich aus Wörtern t bis $t{+}N\text{-}1$.
  - „ich pflücke heute nachmittag wieder ...“ [Beeren],
  - „ich jage heute nachmittag wieder ...“ [Bären]
  - $N{\leq}4$: gleiche Wahrscheinlichkeit für nächstes Wort;
  - $N{\geq}5$: unterschiedliche Wahrscheinlichkeiten.
- Großes N: Wenige Beispiele für jedes N-Gramm, schätzen der Wahrscheinlichkeiten schwierig.
- Kleines N: Viele Beispiele, aber Vorhersage ungenau da verschiedene Wortfolgen als gleich behandelt werden.

Markov-Annahme $(N\text{-}1)$-ter Ordnung

- Wie viele Wahrscheinlichkeiten müssen wir kennen?
  - $N=1$ (nullte Ordnung): ca. 20,000 verschiedene Wortfolgen der Länge 1,
  - $N=2$ (erste Ordnung): ca. 400,000,000,
  - $N=3$ (zweite Ordnung): ca 8×10^{12}.
  - $N=4$ (dritte Ordnung): ca 1.6×10^{17}.
- N-Gramme über Wortstämme, statt über Wörter.
  - Etwas Information geht verloren, aber
  - Deutlich weniger Wahrscheinlichkeiten.

Lernen von N-Gramm-Modellen

- Korpus D ist Folge von Zufallsvariablen w_i; Wertebereich ist das Vokabular v_1 bis v_K.
- Beobachtungen:
  - $x_{v_1 \ldots v_1}, \ldots, x_{v_K \ldots v_K}$: Anzahl der Vorkommen der n-Gramme $v_1 \ldots v_1$ bis $v_K \ldots v_K$ in D.
- Parameter $\boldsymbol{\theta} = (\theta_{v_1|\ldots v_1}, \ldots, \theta_{v_K|\ldots v_K})$

$$\theta_{v_{i_n}|\ldots v_{i_1}} = P(w_n = v_{i_n} \mid w_{n-1} \ldots w_1 = v_{i_{n-1}} \ldots v_{i_1}, \boldsymbol{\theta})$$

- Wie viele Parameter gibt es?
  - P(„ich pflücke Beeren“) = P(„ich“) x P(„pflücke“ | „ich“) x P(„Beeren“ | „ich pflücke“).

Wahrscheinlichkeit eines Satzes

- Wahrscheinlichkeit des Satzes $w_1, \ldots, w_T$ bei gegebenem N-Gramm-Modell:
  - $$P(w_1,...,w_T) = P(w_1)P(w_2 \mid w_1)...P(w_T \mid w_{T-1},...,w_1)$$
    $$= P(w_1)P(w_2 \mid w_1)...P(w_T \mid w_{T-1},...,w_{T-n})$$
    $$= \prod_{i=1}^{N-1} P(w_i \mid w_{i-1},...,w_1) \prod_{i=N}^{T} P(w_i \mid w_{i-1},...,w_{i-n})$$
    $$= \theta_{w_1} \prod_{i=2}^{N-1} \theta_{w_i \mid ...w_1} \prod_{i=N}^{T} \theta_{w_i \mid ...w_{i-n}}$$
- Beispiel: 3-Gramm-Modell:
  - $$P(w_1,...,w_T) = P(w_1)P(w_2 \mid w_1)...P(w_T \mid w_{T-1},...,w_1)$$
    $$= P(w_1)P(w_2 \mid w_1) \prod_{i=3}^{T} P(w_i \mid w_{i-1}, w_{i-2})$$
    $$= \theta_{w_1} \theta_{w_2 \mid w_1} \prod_{i=3}^{T} \theta_{w_i \mid w_{i-1} w_{i-2}}$$

Empirische Inferenz der Modellparameter

- Parameter des Modells sind die echten Auftretenswahrscheinlichkeiten $\boldsymbol{\theta} = (\theta_{v_1|...v_1}, ..., \theta_{v_K|...v_K})$.
- Wahrscheinlichkeiten sind nicht bekannt.
- Echte Wahrscheinlichkeiten sind in einem Korpus von Trainingsdokumenten reflektiert.
- Empirische Inferenz: Berechnung des Posteriors
  - $P(\boldsymbol{\theta} \mid D) \propto P(D \mid \boldsymbol{\theta}) P(\boldsymbol{\theta})$
- MAP-Hypothese: Wahrscheinlichste Parameter gegeben das Korpus D.

Empirische Inferenz der Modellparameter

- Posterior:
  - $P(\boldsymbol{\theta} \mid D) \propto P(D \mid \boldsymbol{\theta}) P(\boldsymbol{\theta})$
- Likelihood: Produkt von Multinomialverteilungen.
  - $P(D \mid \boldsymbol{\theta}) = \prod_{t=1}^{T} P(w_t \mid w_{t-1}, \ldots, w_{t-n+1}, \boldsymbol{\theta})$

    $= \prod_{t=1}^{T} \theta_{w_t \mid w_{t-1}, \ldots, w_{t-n+1}}$

  Spezialfall der Multinomialverteilung: eine Ausführung des Experiments.
- Maximum-Likelihood- (ML-)Schätzer:
  - $\boldsymbol{\theta}^{ML} = \arg\max P(D \mid \boldsymbol{\theta}) = \arg\max P\left(D \mid \begin{pmatrix} \theta_{v_1 \mid \ldots v_1} \\ \ldots \\ \theta_{v_K \mid \ldots v_K} \end{pmatrix} \right)$

    $\theta^{ML}_{v_{i_n} \mid \ldots v_{i_1}} = \arg\max_{\theta_{v_{i_n} \mid \ldots v_{i_1}}} \prod_{t=1}^{T} \theta_{w_t \mid w_{t-1}, \ldots, w_{t-n+1}} = \frac{x_{v_{i_n} \ldots v_{i_1}}}{x_{v_{i_{n-1}} \ldots v_{i_1}}}$

Empirische Inferenz der Modellparameter

- Posterior:
  - $P(\boldsymbol{\theta} \mid D) \propto P(D \mid \boldsymbol{\theta}) P(\boldsymbol{\theta})$
- Prior: Dirichlet, konjugiert zur Multinomialverteilung.
  - $P(\theta) = \prod_{w_{n-1} \ldots w_1 = v_1 \ldots v_1}^{v_K \ldots v_K} P_{Dirichlet}(\theta_{v_1 \mid w_{n-1} \ldots w_1}, \ldots, \theta_{v_K \mid w_{n-1} \ldots w_1} \mid \alpha_{v_1}, \ldots, \alpha_{v_K})$
- Posterior: wieder Dirichletverteilt.
  - $P(\theta \mid D)$

$$= \frac{1}{P(D)} \prod_{t=1}^{T} \theta_{w_t \mid w_{t-1}, \ldots, w_{t-n+1}} \prod_{w_{n-1} \ldots w_1 = v_1 \ldots v_1}^{v_K \ldots v_K} P_{Dirichlet}(\theta_{v_1 \mid w_{n-1} \ldots w_1}, \ldots, \theta_{v_K \mid w_{n-1} \ldots w_1} \mid \alpha_{v_1}, \ldots, \alpha_{v_K})$$

$$= \prod_{w_{n-1} \ldots w_1 = v_1 \ldots v_1}^{v_K \ldots v_K} P_{Dirichlet}\left(\theta_{v_1 \mid w_{n-1} \ldots w_1}, \ldots, \theta_{v_K \mid w_{n-1} \ldots w_1} \mid \alpha_{v_1} + x_{v_1 w_{n-1} \ldots w_1}, \ldots, \alpha_{v_K} + x_{v_K w_{n-1} \ldots w_1}\right)$$

Empirische Inferenz der Modellparameter

- Posterior: wieder Dirichletverteilt.
  - $P(\boldsymbol{\theta} \mid D)$

$$= \prod_{w_{n-1} \ldots w_n = v_1 \ldots v_1}^{v_K \ldots v_K} P_{Dirichlet}\left(\theta_{v_1 \mid w_{n-1} \ldots w_1}, \ldots, \theta_{v_K \mid w_{n-1} \ldots w_1} \mid \alpha_{v_1} + x_{v_1 w_{n-1} \ldots w_1}, \ldots, \alpha_{v_K} + x_{v_K w_{n-1} \ldots w_1}\right)$$

- MAP-Hypothese:
  - $\arg\max_{\theta_{v_{i_n} \mid w_{n-1} \ldots w_1}} P(\boldsymbol{\theta} \mid D)$

$$= \frac{x_{v_{i_n} w_{n-1} \ldots w_1} + \alpha_{v_{i_n}} - 1}{x_{w_{n-1} \ldots w_1} + \sum_{j=1}^{K} (\alpha_{v_j} - 1)}$$

α=2: „Laplace Smoothing“

- Sonderbehandlung $n<N$: Wahrscheinlichkeiten nur aus Satzanfängen schätzen.

Empirische Inferenz der Modellparameter

- Einstellen der Hyperparameter α.
- Wahrscheinlichkeit für ein N-Gramm, das im Trainingskorpus nicht auftaucht:
  - $$\theta_{v_{i_n}|w_{n-1}\ldots w_1} = \frac{0 + \alpha_{v_{i_n}} - 1}{x_{w_{n-1}\ldots w_1} + \sum_{j=1}^{K}(\alpha_{v_j} - 1)}$$
  - Bei einem Wert für alle α: $$\frac{\alpha - 1}{x_{w_{n-1}\ldots w_1} + (\alpha - 1)K}$$

Empirische Inferenz der Modellparameter

- Wie viel Wahrscheinlichkeitsmasse sollte auf ungesehene Kombinationen entfallen?
- Teile Korpus in zwei Teile.
  - Zähle, wie viele Kombinationen $w_n \mid w_{n-1} \ldots w_1$ im zweiten Teil, aber nicht im ersten Teil auftauchen.
  - Stelle dann α so ein, dass $\frac{\alpha - 1}{x_{w_{n-1} \ldots w_1} + (\alpha - 1)K}$ im Durchschnitt über alle Kontexte $w_{n-1} \ldots w_1$ dem gewünschten Verhältnis entspricht.
- Kleine Schwäche des Verfahrens: Im ersten Teil des Korpus kommen nicht mehr N-Gramme vor als im gesamten Korpus.

Wahrscheinlichkeit eines Satzes

- Wahrscheinlichkeit des Satzes $w_1, \ldots, w_T$ bei gegebenem N-Gramm-Modell:
  - $$P(w_1,...,w_T) = P(w_1)P(w_2 \mid w_1)...P(w_T \mid w_{T-1},...,w_1)$$
    $$= P(w_1)P(w_2 \mid w_1)...P(w_T \mid w_{T-1},...,w_{T-n})$$
    $$= \prod_{i=1}^{N-1} P(w_i \mid w_{i-1},...,w_1) \prod_{i=N}^{T} P(w_i \mid w_{i-1},...,w_{i-n+1})$$
    $$= \theta_{w_1} \prod_{i=2}^{N-1} \theta_{w_i \mid ...w_1} \prod_{i=N}^{T} \theta_{w_i \mid ...w_{i-n+1}}$$

- Beispiel: 3-Gramm-Modell:
  - $$P(w_1,...,w_T) = P(w_1)P(w_2 \mid w_1)...P(w_T \mid w_{T-1},...,w_1)$$
    $$= P(w_1)P(w_2 \mid w_1) \prod_{i=3}^{T} P(w_i \mid w_{i-1}, w_{i-2})$$
    $$= \theta_{w_1} \theta_{w_2 \mid w_1} \prod_{i=3}^{T} \theta_{w_i \mid w_{i-1} w_{i-2}}$$

Einstellen von N

- N ist entscheidender Parameter des N-Gramm-Modells.
- Großes N : viele Wahrscheinlichkeiten müssen geschätzt werden.
- Kleines N : zu viel Kontextinformation geht verloren.
- Kompromiss: Interpolation.

Interpolation

- Dem N-Gramm-Sprachmodell liegt ein generatives Prozessmodell zugrunde.
  - Wörter werden iterativ „ausgewürfelt“.
  - Jedes Wort w_i wird nach der Wahrscheinlichkeit $P(w_i \mid w_{i-1},\ldots,w_{i-n+1})$ ausgewürfelt.

- Generatives Modell für interpoliertes N-Gramm:
  - Für jedes Wort w_i wird zunächst Markov-Grad n zwischen 0 und N nach $p(n)$ gezogen.
  - Wort w_i wird dann nach $P(w_i \mid w_{i-1},\ldots,w_{i-n+1})$ gezogen.

Wahrscheinlichkeit eines Satzes

- Wahrscheinlichkeit des Satzes $w_1, \ldots, w_T$ mit interpoliertem N-Gramm-Modell:
  - $$P(w_1,...,w_T) = P(w_1)P(w_2 \mid w_1)...P(w_T \mid w_{T-1},...,w_1)$$
    $$= P(w_1)(p(1)P(w_2) + p(2)P(w_2 \mid w_1))...\sum_{n=1}^{N} p(n)P(w_T \mid w_{T-1},...,w_{T-n+1})$$
    $$= p(w_1)\prod_{i=1}^{N-1}\sum_{n=2}^{i} p_i(n)P(w_i \mid w_{i-1},...,w_1)\prod_{i=N}^{T}\sum_{n=1}^{N} p_i(n)P(w_i \mid w_{i-1},...,w_{i-n+1})$$

- Beispiel: 3-Gramm.
  - $$P(w_1,...,w_T)$$
    $$= P(w_1)(p(1)P(w_2) + p(2)P(w_2 \mid w_1))\prod_{i=3}^{T}\sum_{n=1}^{3} p(n)P(w_i \mid w_{i-1},...,w_{i-n+1})$$

Interpolation

- Likelihood für interpoliertes N-Gramm:
  - $$P(D \mid \boldsymbol{\theta}) = \prod_{t=1}^{T} \sum_{n_t=1}^{N} P(n_t) P(w_t \mid w_{t-1}, \ldots, w_{t-n_t+1}, \boldsymbol{\theta})$$
    $$= \prod_{t=1}^{T} \sum_{n_t=1}^{N} p(n_t) \theta_{w_t \mid w_{t-1}, \ldots, w_{t-n_t+1}}$$
- Posterior:
  - $$P(\boldsymbol{\theta} \mid D)$$
    $$= \prod_{t=1}^{T} \sum_{n_t=1}^{N} p(n_t) \theta_{w_t \mid w_{t-1}, \ldots, w_{t-n_t+1}} \prod_{w_{n-1} \ldots w_1 = v_1 \ldots v_1}^{v_K \ldots v_K} \sum_{n=1}^{N} p(n) P_{Dirichlet}(\theta_{v_1 \mid w_{n-1} \ldots w_1}, \ldots, \theta_{v_K \mid w_{n-1} \ldots w_1} \mid \alpha_{v_1}, \ldots, \alpha_{v_K})$$
    $$= \prod_{w_{n-1} \ldots w_n = v_1 \ldots v_1}^{v_K \ldots v_K} \sum_{n=1}^{N} p(n) P_{Dirichlet}(\theta_{v_1 \mid w_{n-1} \ldots w_1}, \ldots, \theta_{v_K \mid w_{n-1} \ldots w_1} \mid \alpha_{v_1} + x_{v_1 w_{n-1} \ldots w_1}, \ldots, \alpha_{v_K} + x_{v_K w_{n-1} \ldots w_1})$$
- Problem beim Parameterschätzen:
  - n_t ist latente, unbeobachtete Variable.

Interpolation

- Beim Schätzen der Parameter n-Gramm-Modelle müssten wir wissen, welches Wort mit welchem Abhängigkeitswert n erzeugt wurde.
- Nur die Wörter, die mit Wert n generiert wurden sollten in Schätzung des n-Gramm-Modells eingehen.
- Definition Zufallsvariable z_{in}: Wert $z_{in}=1$ zeigt an, dass Wort w_i mit Abhängigkeit n gezogen wurde.
- Werte der z_{in} sind aber unbekannt.

Interpolation

- Problem: Schätzproblem (Schätzen der Paramater $\boldsymbol{\theta} = (\theta_{v_1|\ldots v_1}, \ldots, \theta_{v_K|\ldots v_K})$) mit latenten Variablen z_{in}.
- EM-Algorithmus.
- Idee: Beginne mit zufälligen Parametern.
  - E-Schritt: Berechne Wahrscheinlichkeiten $p(z_{in})$ gegeben die Daten und Parameter.
  - M-Schritt: Schätze $\boldsymbol{\theta} = (\theta_{v_1|\ldots v_1}, \ldots, \theta_{v_K|\ldots v_K})$ basierend auf den aktuellen Werten der $p(z_{in})$.
- Wiederhole bis zur Konvergenz.

Interpolation mit EM, E-Schritt

- $p(z_{in})$: Wahrscheinlichkeit dafür, dass w_i vom n-Gramm-Modell generiert wurde:
  - $p(z_{in}=1) = p(n \mid w_i) \propto p(w_i \mid w_{i-1},\ldots,w_{i-n+1})\,p(n)$

Interpolation mit EM, M-Schritt

- MAP-Schätzer:
  - $\arg\max_{\theta_{v_{i_n}|v_{i_{n-1}}\ldots v_{i_1}}} P(\theta \mid D) = \dfrac{\bar{x}_{v_{i_n}v_{i_{n-1}}\ldots v_{i_1}} + \alpha_{v_{i_n}} - 1}{\bar{x}_{v_{i_{n-1}}\ldots v_{i_1}} - \sum_{i=1}^{K}(\alpha_{v_{i_n}} - 1)K}$
- Zählvariablen:
  - $\bar{x}_{v_{i_n}v_{i\,n-1}\ldots v_{i\,1}} = \sum_{t=1}^{T} p(z_{tn})[[w_t \ldots w_{t-n} = v_{i_n}v_{i\,n-1}\ldots v_{i1}]]$

Anzahl der Parameter → Interpolation

- IBM Tangora:
  - Korpus mit 365 Mio Wörtern, (260.000 Verschiedene)
  - $6.8*10^{10}$ 2-Gramme, 14 Mio davon tauchen auf, 8 Mio nur einmal; 2.2% aller neuen 2-Gramme sind unbekannt.
  - $1.8*10^{16}$ 3-Gramme, 75 Mio tauchen auf, 53 Mio einmal, 14.7% aller neuen 3-Gramme sind unbekannt.

Evaluation von N-Gramm-Modellen

- N-Gramm auf Trainingskorpus gelernt.
- Wie gut ist es?
- Bewertung auf Testkorpus.
- Wie sehr überrascht mich jedes neue Wort des Testkorpus‘?
  - Gutes Modell → weniger Überraschung.
- Entropie H: Wie viele Bit brauche ich um nächstes Wort zu kodieren?
- Perplexität („mittlerer Verzweigungsfaktor“): 2^H.
- Minimum der Entropie = Minimum der Perplexität wenn gelerntes Modell gleich „echtem“ Modell ist.

Evaluation von N-Gramm-Modellen

- N-Gramm-Modell und ein langer Text W.
- Kreuzentropie des Textes gegeben das Modell:

$$H(W \mid m) = -\frac{1}{|W|}\log_2 P_m(W)$$

$$= -\frac{1}{|W|}\log_2 P_m(w_1 \ldots w_n) = -\frac{1}{|W|}\sum_i \log_2 P_m(w_i \mid w_{i-1}, \ldots, w_1)$$

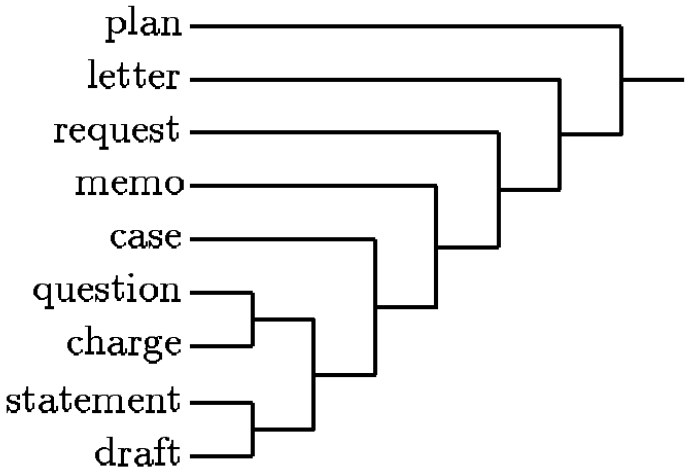
- Perplexität des Modells (durchschnittlicher Verzweigungsfaktor):

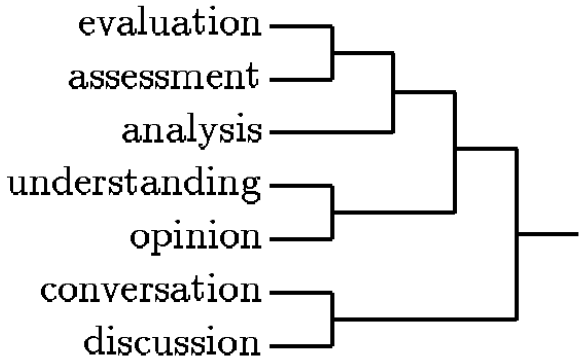
$$PP(W) = 2^{H(W)}$$

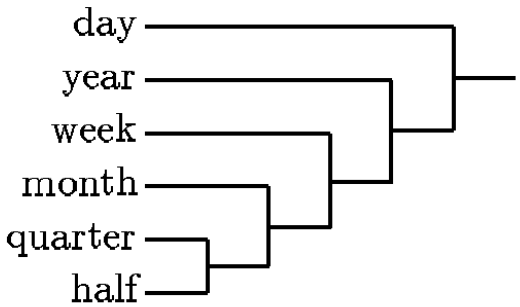
N-Gramm-Klassenmodelle

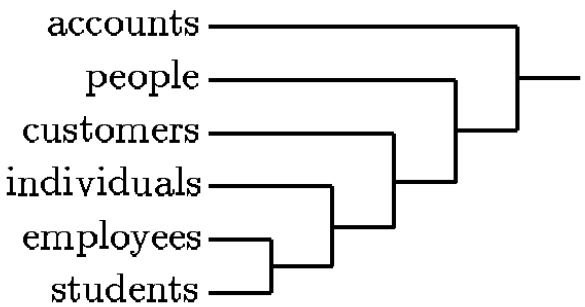
- Manche Wörter haben denselben Kontext.
  - „Wir sehen uns [Montag | Dienstag | ...] Nachmittag“.
- Idee: Wortklassen. $\pi(w)=c$.
- Ein N-Gramm-Modell ist ein Klassenmodell gdw.
- $$P(w_n \mid w_{n-1},\ldots,w_1) = P(w_n \mid c_n)P(c_n \mid c_{n-1},\ldots,c_1)$$
- Klassenmodell hat weniger Parameter (wenn C < V).
- N-Gramm-Klassenmodell = N-Gramm-Modell, wenn jedes Wort eine eigene Klasse hat.
- N-Gramm-Klassenmodell mit einem Wort pro Klasse wird normal aus Trainingskorpus gelernt.
- Dann werden Klassen zusammen gelegt.

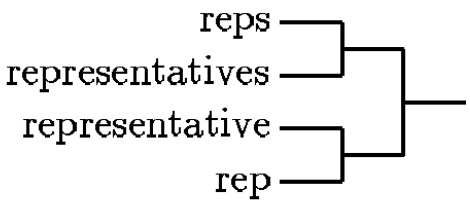
N-Gramm-Klassenmodelle











- Beispiel-Dendrogramme

Beispielaufteilung in 1000 Klassen

Friday Monday Thursday Wednesday Tuesday Saturday Sunday weekends Sundays Saturdays

June March July April January December October November September August

people guys folks fellows CEOs chaps doubters commies unfortunates blokes

down backwards ashore sideways southward northward overboard aloft downwards adrift

water gas coal liquid acid sand carbon steam shale iron

great big vast sudden mere sheer gigantic lifelong scant colossal

man woman boy girl lawyer doctor guy farmer teacher citizen

American Indian European Japanese German African Catholic Israeli Italian Arab

pressure temperature permeability density porosity stress velocity viscosity gravity tension

mother wife father son husband brother daughter sister boss uncle

machine device controller processor CPU printer spindle subsystem compiler plotter

John George James Bob Robert Paul William Jim David Mike

- Beispielklassen

Rechtschreibkorrektur

- Gegeben: Signal $s_1, \ldots, s_T$; gesucht: wahrscheinlichste beabsichtigte Abfolge $w_1, \ldots, w_T$.
- Fehlermodell: $p(s_i \mid w_i)$.
  - Z.B.: $p(s_i \mid w_i) = \exp\left(-\gamma_1 d_{edit}(s_i, w_i) - \gamma_2 d_{phonetisch}(s_i, w_i) - \gamma_3 d_{keyboard}(s_i, d_i)\right)$
- Dekodierung:
  - $$\arg\max_{w_1,\ldots,w_T} p(w_1,\ldots,w_T \mid s_1,\ldots,s_T)$$
    $$= \arg\max_{w_1,\ldots,w_T} p(s_1,\ldots,s_T \mid w_1,\ldots,w_T)\, p(w_1,\ldots,w_T)$$
    $$= \arg\max_{w_1,\ldots,w_T} \prod_{i=1}^{T} p(s_i \mid w_i)\, p(w_i \mid w_{i-1},\ldots,w_{i-n})$$

Rechtschreibkorrektur

- Dekodierung:
  - $\arg\max_{w_1,\ldots,w_T} p(w_1,\ldots,w_T \mid s_1,\ldots,s_T)$

    $= \arg\max_{w_1,\ldots,w_T} \prod_{i=1}^{T} p(s_i \mid w_i) p(w_i \mid w_{i-1},\ldots,w_{i-n})$

- Naive Implementierung:
  - Suche über alle $w_1, \ldots, w_T$ in $O(V^T)$.
- Mit dynamischer Programmierung:
  - Suche über alle $w_1, \ldots, w_T$ in $O(V^N)$.
  - Idee: Tabelle mit $p(w_n \mid \ldots, w_1)$ in $O(V^N)$ vorberechnen.
  - Dann linear durch Text gehen, Für Sprachmodell Werte in Tabelle nachschlagen.