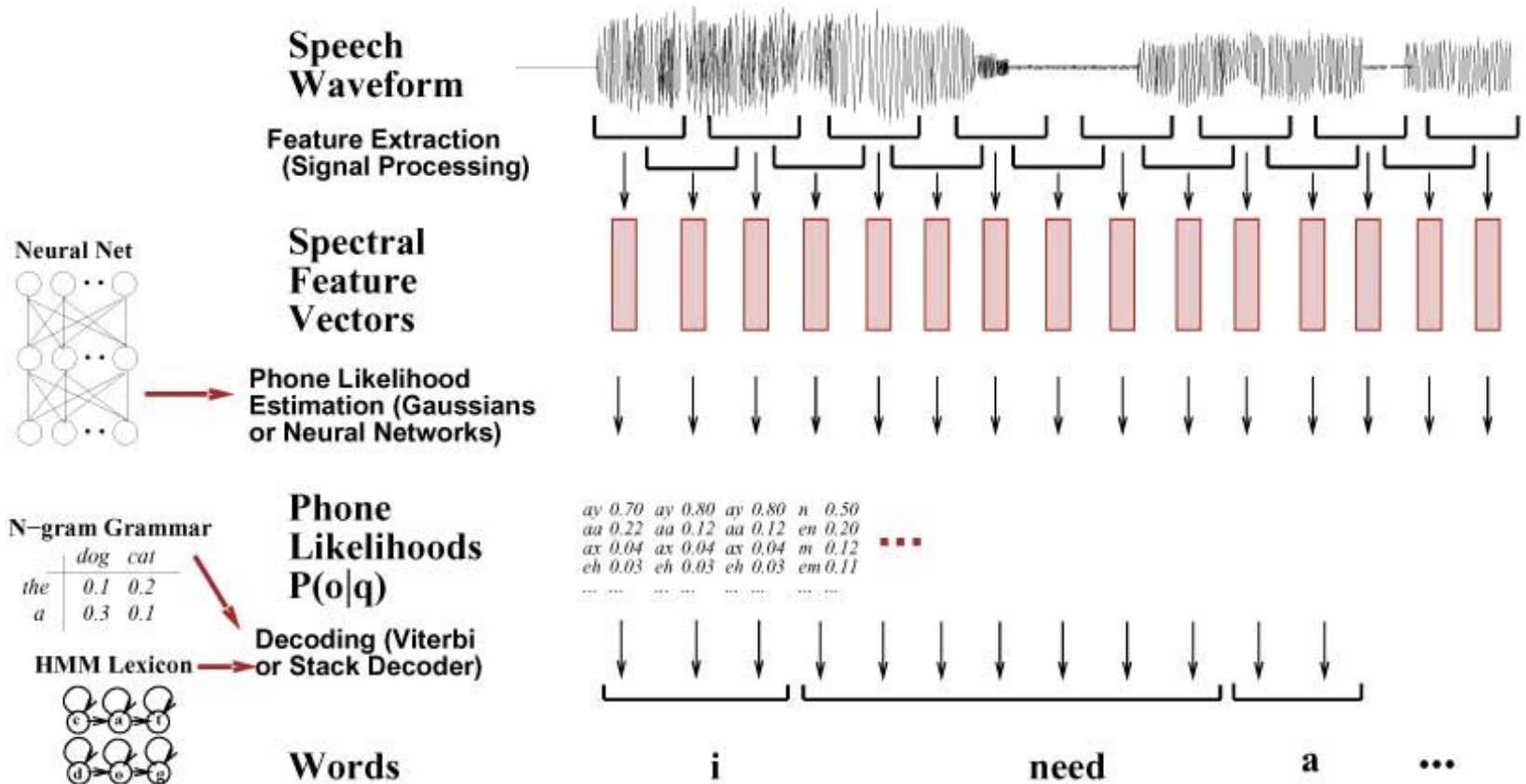


Sprachmodelle

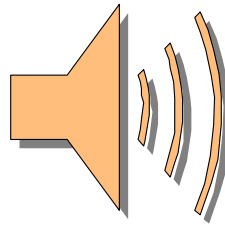
Jan Strunk, Emina Kurtic
10. Juli 2003



Sprachmodell im Spracherkennungssystem

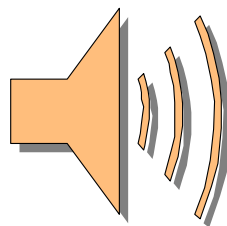


Wozu braucht man ein Sprachmodell?



Wozu braucht man ein Sprachmodell?

I recognize speech.



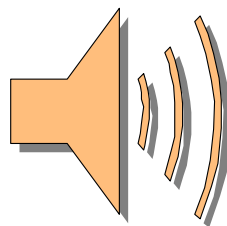
➔ Zwei mögliche Hypothesen eines Spracherkenners:

- ❖ I wreck a nice beach.
- ❖ I recognize speech.



Wozu braucht man ein Sprachmodell?

I recognize speech.



➔ Zwei mögliche Hypothesen eines Spracherkenners

❖ I recognize speech.

~~❖ I wreck a nice beach.~~

Wahrscheinlichere
Hypothese



Sprachmodelle: Formel von Bayes

➔ Das Problem der Spracherkennung

$$\operatorname{argmax} P(W | A) = \frac{P(A | W) \cdot P(W)}{P(A)}$$



Sprachmodelle: Formel von Bayes

➔ Das Problem der Spracherkennung

$$\operatorname{argmax} P(W | A) = \frac{P(A | W) \cdot P(W)}{P(A)}$$

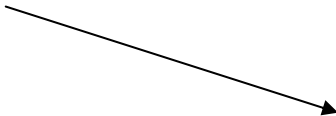
Akustische Daten sind gegeben. Ihre Wahrscheinlichkeit kann außer Acht gelassen werden.



Sprachmodelle: Formel von Bayes

➔ Das Problem der Spracherkennung

Akustisches Modell


$$\operatorname{argmax} P(W | A) = \frac{P(A | W) \cdot P(W)}{P(A)}$$



Sprachmodelle: Formel von Bayes

➔ Das Problem der Spracherkennung

Linguistisches Modell
apriori-Wahrscheinlichkeit

$$\operatorname{argmax} P(W | A) = \frac{P(A | W) \cdot P(W)}{P(A)}$$



Sprachmodelle: Aufgabe

- ➔ Die Aufgabe des Sprachmodells besteht darin:
 - ❖ Die Wahrscheinlichkeit einer Sequenz von Wörtern zu berechnen,
 - ❖ Den Suchraum für mögliche folgende Wörter einzuschränken,
 - ❖ In Kombination mit dem akustischen Modell die Wahrscheinlichkeit der Hypothesen zu berechnen.

- ➔ Die Struktur der Sprache oberhalb der Wortebene als linguistisches Wissen für die Spracherkennung auszunutzen.



Sprachmodelle: Realisierung

- ➔ Sprachmodelle: Regelsysteme, die die Struktur der Sprache beschreiben

- ➔ Implementierungsmöglichkeiten:
 - ❖ Deterministische Grammatiken
 - ❖ N-Gramm-Modelle (Markov-Modelle)
 - ❖ Probabilistische kontextfreie Grammatiken



N-Gramme

- ➔ Die Wahrscheinlichkeit des i-ten Wortes wird in Abhängigkeit vom Kontext aus den i-1 vorhergehenden Wörtern berechnet.

$$P(w_i | w_1, \dots, w_{i-1})$$

- ➔ Markov-Annahme: $n = 2, 3$

- ❖ $n=2$ (Bigramme): $P(w_i | w_1, \dots, w_{i-1}) \approx P(w_i | w_{i-1})$

- ❖ $n=3$ (Trigramme): $P(w_i | w_1, \dots, w_{i-1}) \approx P(w_i | w_{i-2}, w_{i-1})$



Das Trainingskorpus

➔ Korpus: Eine Sammlung von sprachlichen Daten.

➔ In unserem Falle:

- ❖ Ein 6 MB-Ausschnitt aus dem *Neue Zürcher Zeitung*-Korpus.
- ❖ In Sätze aufgeteilt und durchgängig klein geschrieben.



Token vs. Type

- ➔ Token: Ein einzelnes sprachliches Datum in einem Korpus, ein einzelnes Wort.
- ➔ Type: Ein Wort als abstrakte Klasse, die in mehreren Instanzen in einem Korpus vorkommen kann.

- ➔ Der folgende Text:

*jetzt versucht die **gleiche** gruppe das **gleiche** ziel in teilschritten zu erreichen .*

enthält 12 Token und 11 Types.



Korpusgröße vs. Größe des Vokabulars

➔ Korpusgröße N : Anzahl der Token im Korpus

➔ Vokabular V : Anzahl der Types im Korpus



Training des Modells

- ➔ Ableitung eines statistischen Modells aus dem Trainingskorpus (Training)
- ➔ Das heißt zunächst einfach: Zählen!
 - ❖ Anfügen von $n-1$ Dummytoken an den Anfang des Korpus oder der einzelnen Sätze (z.B. <START>)
 - ❖ Ermittlung der absoluten Häufigkeiten aller $1 \dots n$ -Gramme im Trainingskorpus



Das einfachste Modell: MLE

➔ Maximum Likelihood Estimation

- ❖ Wahrscheinlichkeit eines N-Gramms ist gleich der relativen Häufigkeit im Trainingskorpus

$$P(w_1, \dots, w_n) = \frac{H(w_1, \dots, w_n)}{N_{n\text{-Gramme}}}$$



Das einfachste Modell: MLE

➔ Beispiel 1

- ❖ Unser Trainingskorpus enthält insgesamt 899053 Bigramm-Token.
- ❖ Das Bigramm „die schweiz“ kommt 1009 mal vor.

$$P_{MLE}(\text{die schweiz}) = \frac{H(\text{die schweiz})}{N_{\text{Bigramme}}} = \frac{1009}{899503} \approx 0,00112173$$



Das einfachste Modell: MLE

➔ Maximum Likelihood Estimation

- ❖ Die bedingte Wahrscheinlichkeit für w_n gegeben den vorhergehenden Kontext w_1, \dots, w_{n-1} ist

$$P(w_n \mid w_1, \dots, w_{n-1}) = \frac{H(w_1, \dots, w_n)}{H(w_1, \dots, w_{n-1})}$$



Das einfachste Modell: MLE

➔ Beispiel 2

- ❖ Das Bigramm „die schweiz“ kommt 1009 mal vor.
- ❖ Das Unigramm „die“ kommt 32491 mal vor.

$$P_{MLE}(\text{schweiz}|\text{die}) = \frac{H(\text{die schweiz})}{H(\text{die})} = \frac{1009}{32491} \approx 0,03105475$$



Die Wahrscheinlichkeit einer Sequenz

➔ Produkt der einzelnen bedingten Wahrscheinlichkeiten

$$P(w_1, \dots, w_i) = P(w_1 | w_0 \dots w_{1-n+1}) \cdot \dots \cdot P(w_i | w_{i-1}, \dots, w_{i-n+1})$$

➔ Beispiel 3

$$P(\text{die schweiz ist sehr gut .}) = P(\text{die} | \langle \text{START} \rangle) \cdot P(\text{schweiz} | \text{die}) \cdot P(\text{ist} | \text{schweiz}) \\ \cdot P(\text{sehr} | \text{ist}) \cdot P(\text{gut} | \text{sehr}) \cdot P(\text{.} | \text{gut})$$

$$= 4573/37348 \cdot 1009/32491 \cdot 62/2626 \cdot 7/7643 \cdot 9/418 \cdot 15/308$$

$$\approx 8,6218 \text{ e}^{-11}$$



„Das Problem der knappen Daten“

- ➔ Problem 1: unbekannte Wörter
- ➔ Problem 2: prinzipiell mögliche, aber zufällig im Trainingskorpus nicht vorkommende n-Gramme
- ➔ Das MLE-Modell weist wohlgeformten Sätzen die Wahrscheinlichkeit 0 zu.



Anzahl der Parameter eines n-Gramm-Modells

- ➔ Die Anzahl der möglichen n-Gramm-Typen in einem Markov-Modell wächst exponentiell zur Ordnung des Modells.

n	Vokabular	Mögliche n-Gramme
2	20000	$20000^2 = 4 \cdot 10^8$
3	20000	$20000^3 = 8 \cdot 10^{12}$
4	20000	$20000^4 = 16 \cdot 10^{16}$

- ➔ Beschränkung auf Bigramm- und Trigramm-Modelle und ein möglichst kleines Vokabular!



Einige Daten zum Trainingskorpus

Das Korpus enthält 936.401 Token und 61.463 verschiedene Typen.
Davon kommen 32.628 nur einmal vor (53.0855962123554 %).

Es enthält 899.053 Bigramme und 379.208 verschiedene Bigramm-Typen.
Davon kommen 298.984 nur einmal vor (78.8443281787304 %).
Von den möglichen 3.777.700.369 Bigramm-Typen kommen nur
0.0100380645090807 % vor.

Es enthält 861705 Trigramme und 665892 verschiedene Trigramm-Typen.
Davon kommen 607015 nur einmal vor (91.1581758002799 %).
Von den möglichen 232.188.797.779.847 Trigramm-Typen kommen nur
2.86789029603131e-007 % vor.



„Das Problem der knappen Daten“

➔ Eigenschaften natürlicher Sprache

❖ Zipf'sches Gesetz:

- Sehr wenige sehr, sehr häufige Worttypen
- Sehr viele sehr, sehr seltene Worttypen

❖ Mangelnde Zeitinvarianz

➔ Auch immer größere Trainingskorpora helfen nicht gegen das Problem der knappen Daten!



Glätten des Sprachmodells

- ➔ Ableitung eines MLE-Modells aus dem Trainingskorpus
- ➔ Umverteilung von Wahrscheinlichkeitsmasse von den vorgekommenen n-Grammen an die nicht vorgekommenen n-Gramme
- ➔ Auch nicht im Trainingskorpus vorgekommene n-Gramme erhalten eine Wahrscheinlichkeit > 0 .



Add-One-Glättung (Laplace)

- ➔ Man addiert zur absoluten Häufigkeit aller n-Gramm-Typen den Wert 1 → keine Nullen mehr!
- ➔ Die bedingte Wahrscheinlichkeit ist dann:

$$P(w_n | w_1, \dots, w_{n-1}) = \frac{H(w_1, \dots, w_n) + 1}{H(w_1, \dots, w_{n-1}) + V}$$

Vokabular V: Anzahl der Types im Korpus



Add-One-Glättung (Laplace)

- ➔ Add-One-Glättung ist ein sehr einfaches, aber auch ein sehr schlechtes Verfahren
 - ❖ Vergibt zuviel Wahrscheinlichkeitsmasse an ungesehene Ereignisse.
 - ❖ Weist allen nicht aufgetretenen n-Grammen die gleiche Wahrscheinlichkeit zu.
- ➔ In der Praxis werden bessere Glättungsverfahren benutzt: Witten-Bell-Glättung, Good-Turing-Glättung



Back-Off-Modelle

➔ Eine Alternative sind Back-Off-Modelle

- ❖ Benutzung kürzerer Kontexte, wenn längere nicht im Trainingskorpus vorgekommen sind.

➔ Katz' Back-Off

$$P(w_i | w_{i-2}, w_{i-1}) = \begin{cases} P(w_i | w_{i-2}, w_{i-1}) & \text{wenn } H(w_{i-2}, w_{i-1}, w_i) > 0 \\ \alpha_1 P(w_i | w_{i-1}) & \text{wenn } H(w_{i-2}, w_{i-1}, w_i) = 0 \\ & \text{und } H(w_{i-1}, w_i) > 0 \\ \alpha_2 P(w_i) & \text{andernfalls} \end{cases}$$



Back-Off-Modelle

- ➔ Back-Off-Modelle müssen mit Glättungs-Verfahren kombiniert werden.
 - ➔ Verteilung von Wahrscheinlichkeitsmasse an Modelle niedrigerer Ordnung.
- ➔ Wahrscheinlichkeit der Modelle niedrigerer Ordnung muss skaliert werden (α_1, α_2).



Was modellieren N-Gramm-Modelle?

- ➔ N-Gramm-Modelle sind nur eine Möglichkeit
 - ❖ Sie modellieren
 - Lokale syntaktische
 - Lokale semantische
 - Und einfache Kookurrenz-Beziehungen.
 - ❖ Sie modellieren nicht
 - Nicht-lokale syntaktische
 - Nicht-lokale semantische
 - Sonstige nicht-lokale Abhängigkeiten.



Was modellieren N-Gramm-Modelle nicht?

➔ Beispiel: lange syntaktische Abhängigkeit

❖ Er **hat** ihm vor drei Tagen das Buch **gegeben**.

vs.

❖* Er **hat** ihm vor drei Tagen das Buch **gab**.

➔ Dafür sind andere formale Grammatiktypen besser geeignet (z.B. Probabilistic Context Free Grammar)



Warum dann N-Gramm Modelle?

- ➔ Sie sind einfach automatisch aus einem Trainingskorpus ableitbar.
- ➔ Sie haben sich im Einsatz in kompletten Spracherkennungssystemen bewährt.
- ➔ Handgeschriebene formale Grammatiken erreich(t)en nicht die nötige Abdeckung.



Lesestoff

Carstensen, K.-U. et al. (Hrsg.) (2001): **Computerlinguistik und Sprachtechnologie. Eine Einführung.** Spektrum, Heidelberg.

Chen, Stanley F.; Goodman, Joshua (1998): **An Empirical Study of Smoothing Techniques in Language Modelling.** Technical Report TR-10-98, Center for Research in Computing Technology, Harvard University.

Jurafsky, Daniel; Martin, James H. (2000): **Speech and Language Processing.** Prentice-Hall, Upper Saddle River. **Kap. 6, 7.**

Manning, Christopher D.; Schütze, Hinrich (1999): **Foundations of Statistical Natural Language Processing.** MIT Press, Cambridge. **Kap. 6, 9, 11.**



Internetadresse

- ➔ Die Folien, das Korpus und die Programme sind unter folgender Adresse im Internet zu finden:

www.linguistics.ruhr-uni-bochum.de/~strunk/sprachmodelle/

