# N - Gramm

## Sprachmodelle

Felix Pötzsch, 580106

Timofei Borisov, 580092

Simon Furitsch, 578153

# Einführung:

- Was ist ein Sprachmodell:
  - Sprachmodelle sind der Versuch z.B. natürliche Sprache als mathematisches Modell repräsentieren zu können.
  - Ihnen liegt meist ein stochastischer Prozess zugrunde

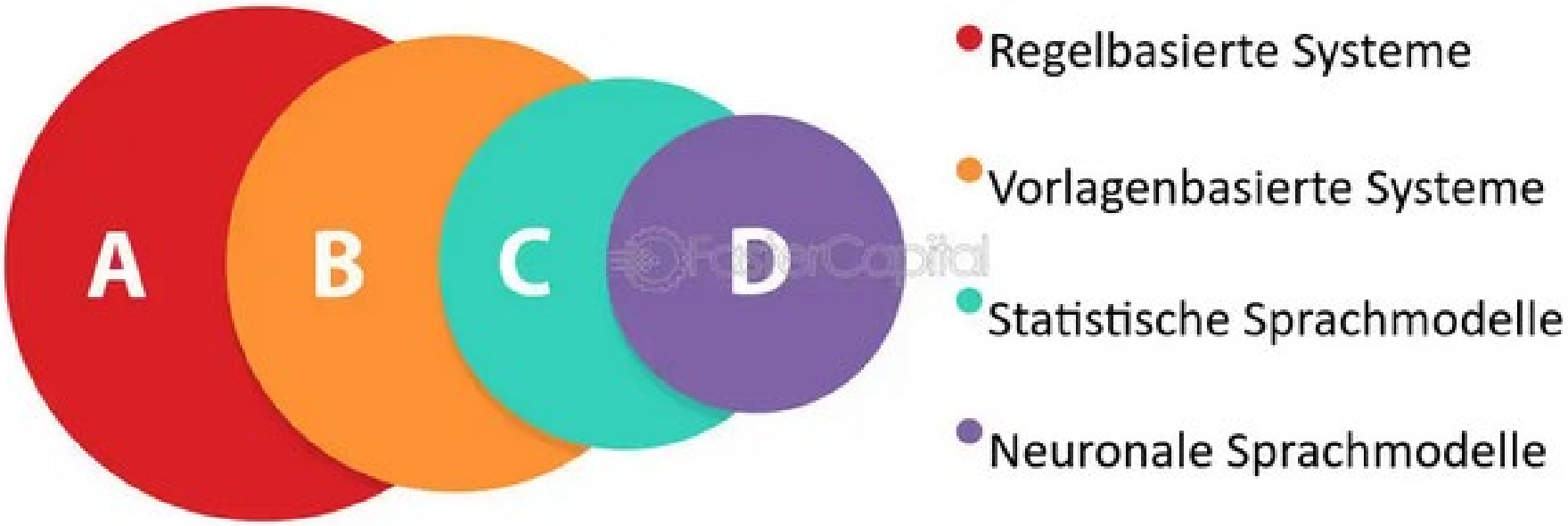
  - Kenntnisse stehen im Voraus zur Verfügung
    - Wissen über Sprache im Allgemeinen
    - Wissen über die Kommunikationssituation
  - Bestehen meist aus mehreren Teilmodellen
    - Vokabular
    - Häufigkeit der Wörter
    - Satzgrammatik

# Einführung:

- Typisch gebräuchliche Sprachmodelle:
  - RNN (Rekurrente Neuronale Netze)
  - LSTM (Long Short-Term Memory Netze)
  - GRU (Gated Recurrent Units)
  - Transformer Modelle:
    - GPT (Generative Pre-Trained Transformer)
    - GPT-3 ((Generative Pre-Trained Transformer 3)

# Einführung:

N-Gramm




*fastercapital.com/de/keyword/neuronale-sprachmodelle.html*

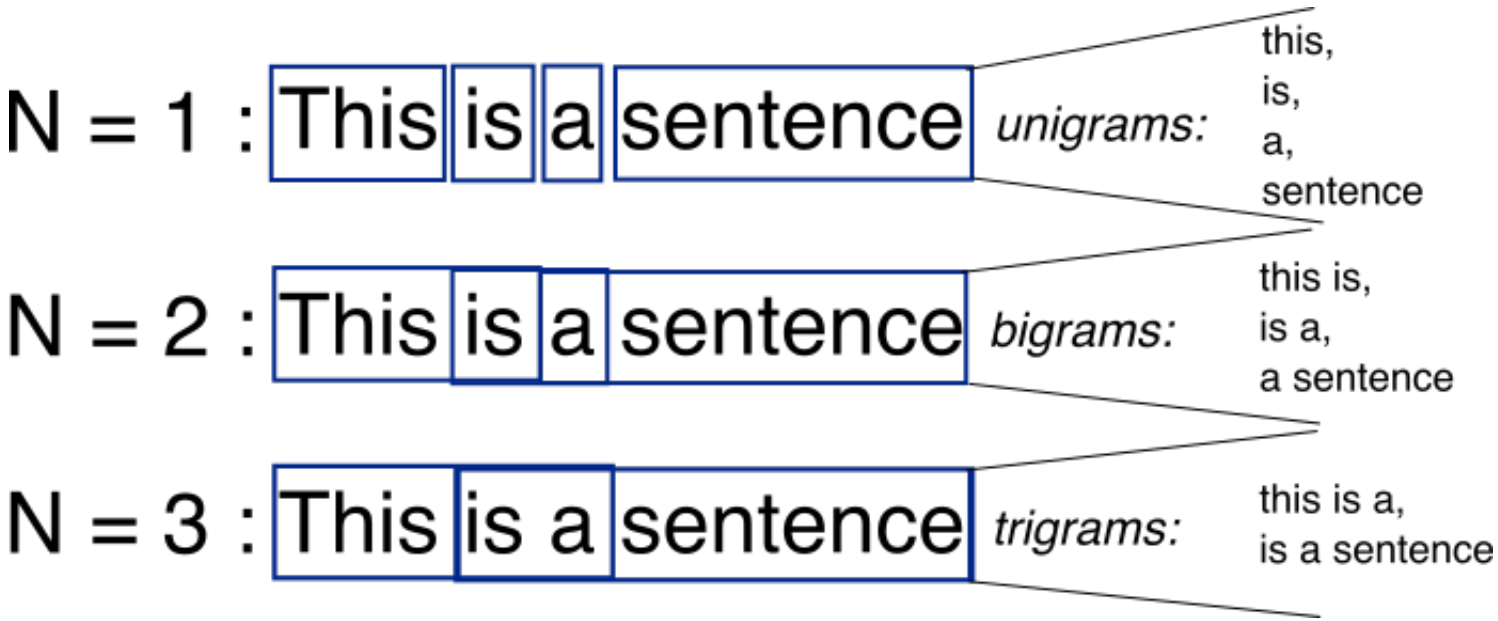
# Einführung:

- N-Gramm ist:
  - In Linguistik eine Sequenz von N aufeinanderfolgenden Teilen (Fragmente) eines Textes.
- Es gibt:
  - Unigramme, Bigramme, Trigramme, … , Multigramme
- Können sein:
  - Buchstaben, Silben, *Laute*, Wörter

# Einführung:

- Typische Anwendungsgebiete sind:
  - Rechtschreibkorrektur
  - Textvervollständigung (Wörter Vorhersage)
  - Textanalyse (Beurteilung)
  - Spracherkennung
  - Handschrifterkennung

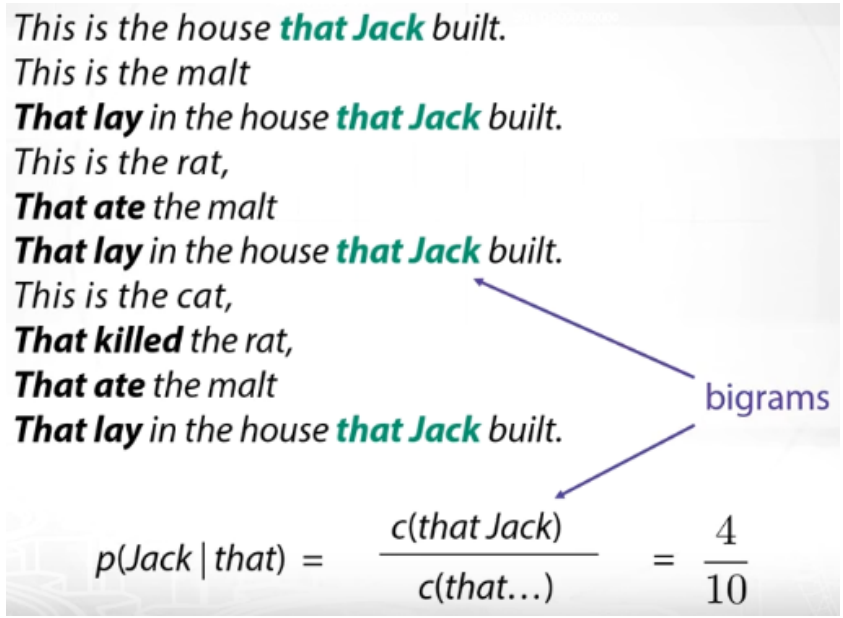
# Einführung:

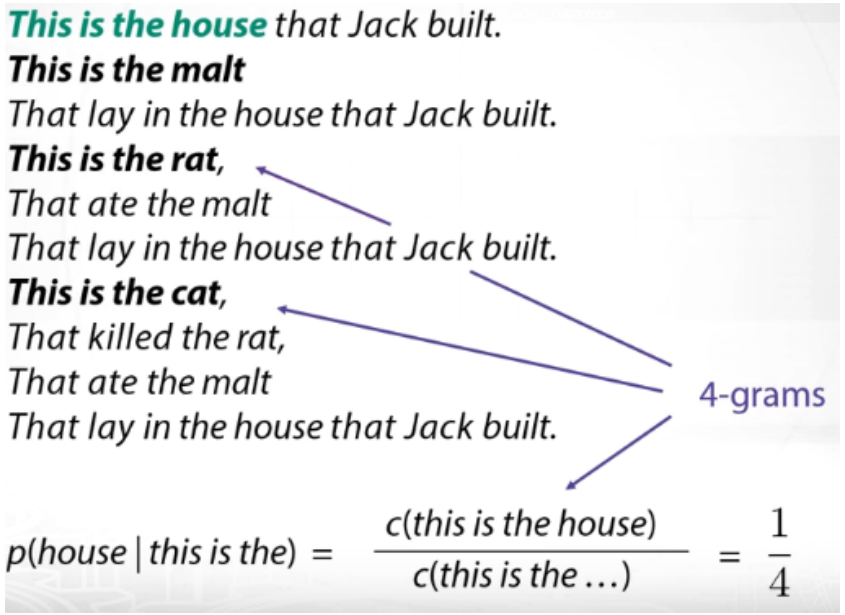



*deepai.org/machine-learning-glossary-and-terms/n-gram*

# Einführung:

$$p(house \mid this\ is\ the) = ?$$




*NLP: Understanding the N-gram language models*
*https://www.youtube.com/watch?v=GiyMGBuu45w&t=44s*

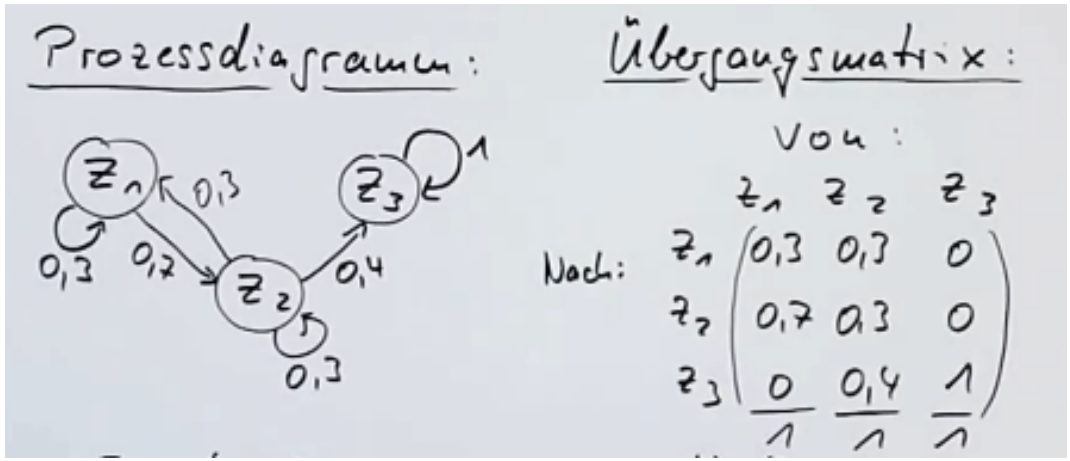
# Einführung:

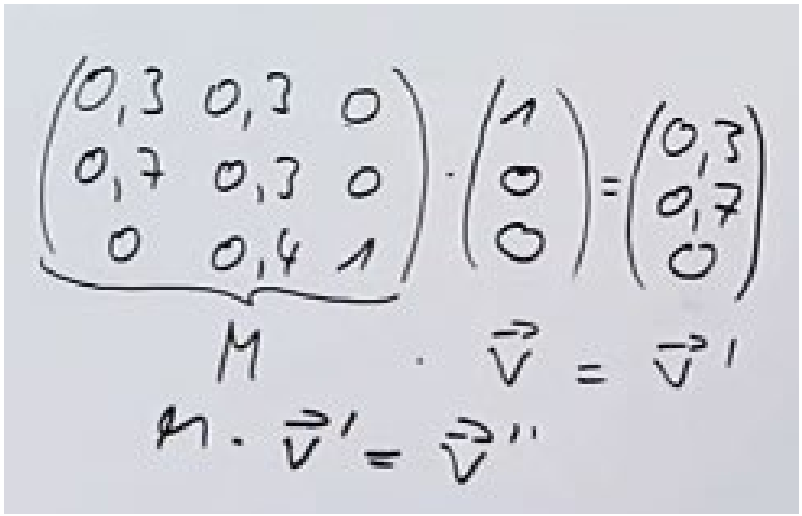

https://kapoole.github.io/JaneAustenandMe/ Trainiert mit den Büchern: 'Sense and Sensibility', 'Persuasion', und 'Emma'.

# Einführung:

- Was wird benötigt:
  - Daten (Text)
  - Wahrscheinlichkeiten für Wörter
  - Markov Annahme
    - Konstante Beobachtungen
    - Wechsel zwischen endlich vielen Zustäden
    - Kümmere dich nicht um alle Wörter, sondern nur um z.B die letzten Paar

# Einführung:



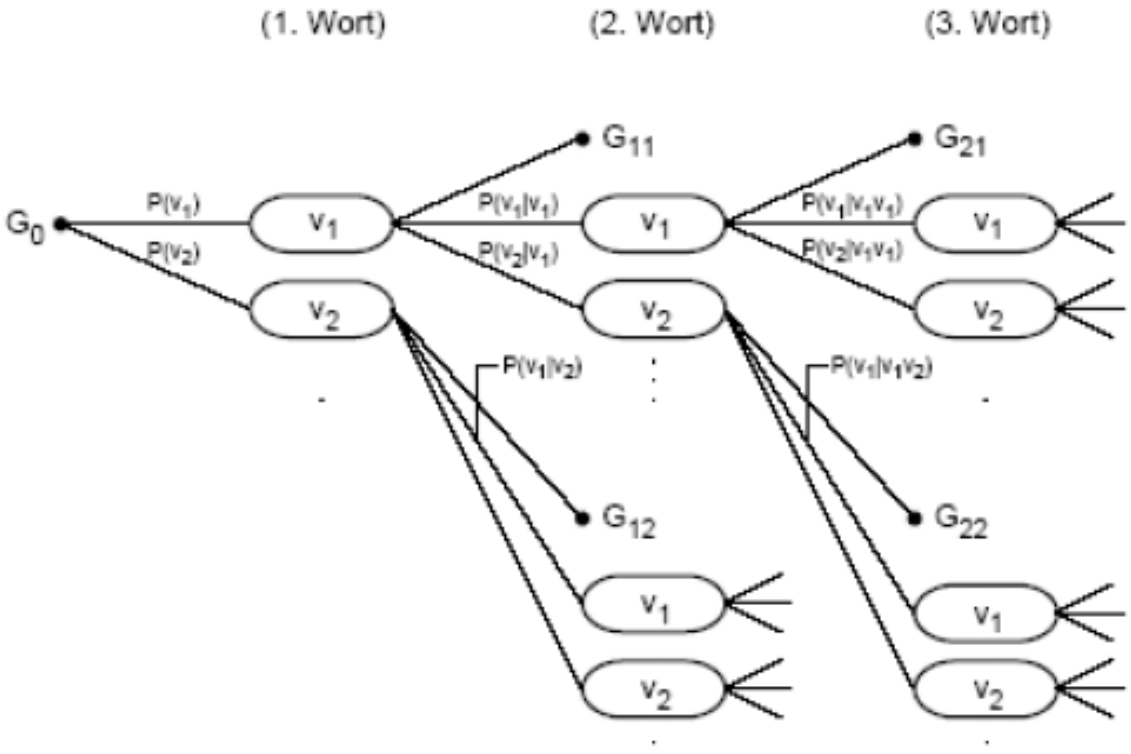




| | Italien | Deutschland | Violinist | Pianist |
|---|---|---|---|---|
| $W_1$ = Beethoven | 0 | 1 | 0 | 1 |
| $W_2$ = Paganini | 1 | 0 | 1 | 0 |

*Markoff Kette, Markov Kette, Übergangsprozess, stochastischer Prozess | Mathe by Daniel Jung: https://www.youtube.com/watch?v=NKoPwE2LQhc*

# Einführung:

- Ein paar Eigenheiten:
  - Zu großes N
    - Kann zu Totalversagen führen
  - Größerer Korpus besser
    - Aber auch da nicht alles finden
    - Prefixproblem




*Technische Universität Chemnitz: Sprachmodellierung Handout PDF: https://www.tu-chemnitz.de/informatik/KI/edu/spraver/ws2013/sprach_7.pdf*

# Einführung:

$$P(t) = P(w_1 w_2 \dots w_n) = P(w_1)P(w_2|w_1)\dots P(w_n|w_1,\dots w_{n-1})$$
$$= \prod_{i=1}^{n} P(w_i|w_1 \dots w_{i-1})$$

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

| Rang | Wortpaar | Häufigkeit in % |
|---|---|---|
| 1 | *in der* | 0.2849 |
| 2 | *bei der* | 0.2398 |
| 3 | *für die* | 0.1945 |
| 4 | *in den* | 0.1419 |
| 5 | *und der* | 0.1377 |
| 6 | *und die* | 0.1282 |
| 7 | *das ist* | 0.1233 |
| 8 | *auf die* | 0.1022 |
| 9 | *von der* | 0.1015 |
| 10 | *mit dem* | 0.0970 |
| 11 | *mit der* | 0.0867 |
| 12 | *in die* | 0.0818 |
| 13 | *dass die* | 0.0746 |
| 14 | *ist die* | 0.0725 |
| 15 | *es ist* | 0.0721 |

*Institut für Computerlinguistik Uni Heidelberg – N-Modelling, Katja Markert*:
*https://www.cl.uni-heidelberg.de/courses/ws19/ecl/material/ngram_handout.pdf*

*Technische Universität Chemnitz: Sprachmodellierung Handout PDF:*
*https://www.tu-chemnitz.de/informatik/KI/edu/spraver/ws2013/sprach_7.pdf*

# Einführung:

- Probleme:
  - Analysiert und generiert Text auf Basis von statistischen Mustern
  - Mangelt also häufig an semantischem Verständnis
  - Einschränkungen insbesondere auch wenn es um langfristige Abhängigkeiten (Kohärenz) und Kontexte geht.
  - UNK (Unknown Words/Unseen N-Gramm)
  - Underflow
  - Nullwahrscheinlichkeiten

- *Was ist trotzdem gut:*
  - *Effizienz*
  - *Für beliebige Sprachen einsetzbar (kein linguistisches Wissen nötig)*
  - *Günstig zu trainieren*

# Einführung:

- Wie also besser werden:
  - Information
    - Anpassen
    - Auswahl
  - Smoothing - Reevaluation von Nullwahrscheinlichkeit und niedrigen Wahrschinlichkeiten
    - Höhere Werte zuweisen damit sie auch vorkommen können
  - Perplexität
    - Evaluation

# Was sind un-seen N-Gramme?

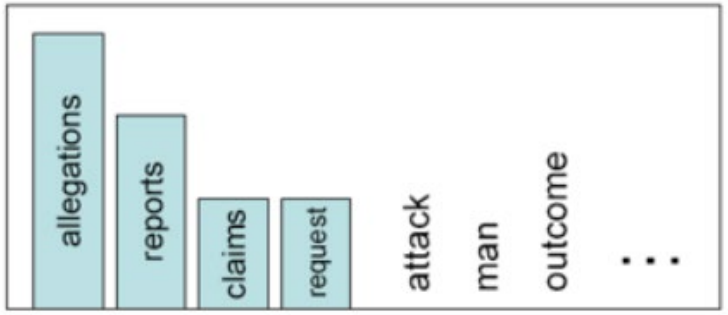
“Data Sparsity”

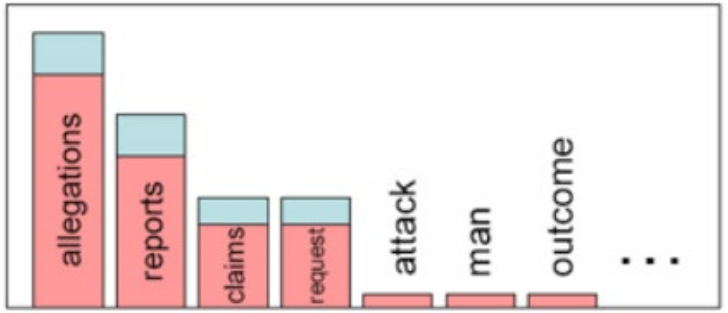
| | i | want | to | eat | chinese | food | lunch | spend |
|---|---|---|---|---|---|---|---|---|
| **i** | 5 | 827 | 0 | 9 | 0 | 0 | 0 | 2 |
| **want** | 2 | 0 | 608 | 1 | 6 | 6 | 5 | 1 |
| **to** | 2 | 0 | 4 | 686 | 2 | 0 | 6 | 211 |
| **eat** | 0 | 0 | 2 | 0 | 16 | 2 | 42 | 0 |
| **chinese** | 1 | 0 | 0 | 0 | 0 | 82 | 1 | 0 |
| **food** | 15 | 0 | 15 | 0 | 1 | 4 | 0 | 0 |
| **lunch** | 2 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| **spend** | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |

*Dan Jurafsky. Speech and Language Processing*
*An Introduction to Natural Language Processing,*
*Computational Linguistics, and Speech Recognition*

# Smoothing oder Discounting Techniken:

1. Laplace Smoothing.
2. Good-Turing Smoothing.
3. Katz Backoff Smoothing.




*Source: https://komputasi.wordpress.com/wp-content/uploads/2019/03/husni-teknik-estimasi-dalam-pemrosesan-bahasa-alami.pdf*

# Laplace Smoothing:

| | i | want | to | eat | chinese | food | lunch | spend |
|---|---|---|---|---|---|---|---|---|
| i | 5 | 827 | 0 | 9 | 0 | 0 | 0 | 2 |
| want | 2 | 0 | 608 | 1 | 6 | 6 | 5 | 1 |
| to | 2 | 0 | 4 | 686 | 2 | 0 | 6 | 211 |
| eat | 0 | 0 | 2 | 0 | 16 | 2 | 42 | 0 |
| chinese | 1 | 0 | 0 | 0 | 0 | 82 | 1 | 0 |
| food | 15 | 0 | 15 | 0 | 1 | 4 | 0 | 0 |
| lunch | 2 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| spend | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |

| | i | want | to | eat | chinese | food | lunch | spend |
|---|---|---|---|---|---|---|---|---|
| i | 6 | 828 | 1 | 10 | 1 | 1 | 1 | 3 |
| want | 3 | 1 | 609 | 2 | 7 | 7 | 6 | 2 |
| to | 3 | 1 | 5 | 687 | 3 | 1 | 7 | 212 |
| eat | 1 | 1 | 3 | 1 | 17 | 3 | 43 | 1 |
| chinese | 2 | 1 | 1 | 1 | 1 | 83 | 2 | 1 |
| food | 16 | 1 | 16 | 1 | 2 | 5 | 1 | 1 |
| lunch | 3 | 1 | 1 | 1 | 1 | 2 | 1 | 1 |
| spend | 2 | 1 | 2 | 1 | 1 | 1 | 1 | 1 |

*Source: Dan Jurafsky. Speech and Language Processing An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*

# Laplace Smoothing:

| | i | want | to | eat | chinese | food | lunch | spend |
|---|---|---|---|---|---|---|---|---|
| i | 6 | 828 | 1 | 10 | 1 | 1 | 1 | 3 |
| want | 3 | 1 | 609 | 2 | 7 | 7 | 6 | 2 |
| to | 3 | 1 | 5 | 687 | 3 | 1 | 7 | 212 |
| eat | 1 | 1 | 3 | 1 | 17 | 3 | 43 | 1 |
| chinese | 2 | 1 | 1 | 1 | 1 | 83 | 2 | 1 |
| food | 16 | 1 | 16 | 1 | 2 | 5 | 1 | 1 |
| lunch | 3 | 1 | 1 | 1 | 1 | 2 | 1 | 1 |
| spend | 2 | 1 | 2 | 1 | 1 | 1 | 1 | 1 |

Unterschiedliche Wörter. V = 1446

$$P(w_i) = \frac{c_i}{N}$$

$$P_{\text{Laplace}}(w_i) = \frac{c_i + 1}{N + V}$$

$$P(to \mid want) = \frac{608}{927} = 0.66$$

$$P_L(to \mid want) = \frac{608 + 1}{927 + 1446} = 0.26$$

*Dan Jurafsky. Speech and Language Processing*
*An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*

*Prof. Dr. Johannes Maucher*
*https://griesshaber.pages.mi.hdm-stuttgart.de/nlp/04ngram/01ngram.html, Accessed: June 1, 2024.*

# Laplace Smoothing:

| | i | want | to | eat | chinese | food | lunch | spend |
|---|---|---|---|---|---|---|---|---|
| i | 6 | 828 | 1 | 10 | 1 | 1 | 1 | 3 |
| want | 3 | 1 | 609 | 2 | 7 | 7 | 6 | 2 |
| to | 3 | 1 | 5 | 687 | 3 | 1 | 7 | 212 |
| eat | 1 | 1 | 3 | 1 | 17 | 3 | 43 | 1 |
| chinese | 2 | 1 | 1 | 1 | 1 | 83 | 2 | 1 |
| food | 16 | 1 | 16 | 1 | 2 | 5 | 1 | 1 |
| lunch | 3 | 1 | 1 | 1 | 1 | 2 | 1 | 1 |
| spend | 2 | 1 | 2 | 1 | 1 | 1 | 1 | 1 |

| | i | want | to | eat | chinese | food | lunch | spend |
|---|---|---|---|---|---|---|---|---|
| i | 0.0015 | 0.21 | 0.00025 | 0.0025 | 0.00025 | 0.00025 | 0.00025 | 0.00075 |
| want | 0.0013 | 0.00042 | 0.26 | 0.00084 | 0.0029 | 0.0029 | 0.0025 | 0.00084 |
| to | 0.00078 | 0.00026 | 0.0013 | 0.18 | 0.00078 | 0.00026 | 0.0018 | 0.055 |
| eat | 0.00046 | 0.00046 | 0.0014 | 0.00046 | 0.0078 | 0.0014 | 0.02 | 0.00046 |
| chinese | 0.0012 | 0.00062 | 0.00062 | 0.00062 | 0.00062 | 0.052 | 0.0012 | 0.00062 |
| food | 0.0063 | 0.00039 | 0.0063 | 0.00039 | 0.00079 | 0.002 | 0.00039 | 0.00039 |
| lunch | 0.0017 | 0.00056 | 0.00056 | 0.00056 | 0.00056 | 0.0011 | 0.00056 | 0.00056 |
| spend | 0.0012 | 0.00058 | 0.0012 | 0.00058 | 0.00058 | 0.00058 | 0.00058 | 0.00058 |

*Dan Jurafsky. Speech and Language Processing*
*An Introduction to Natural Language Processing,*
*Computational Linguistics, and Speech Recognition*

# Good-Turing-Smoothing:

- 10 red balls,
- 3 green balls,
- 2 blue balls
- 1 black ball
- 1 brown ball
- 1 grey ball

Wie groß ist die Wahrscheinlichkeit, dass die nächste Kugel schwarz ist?

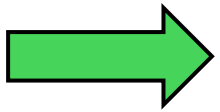
$$P(black) = \frac{1}{18}.$$

Wie groß ist die Wahrscheinlichkeit, dass die nächste Kugel eine bisher ungesehene Farbe hat?

$$P_{GT}(unseen) = \frac{N_1}{Z}$$

N - gibt die Anzahl der Arten an, die bisher gesehen gesehen wurden.
(N1 - Ein mal gesehen)

Z - die Anzahl der bisherigen Beobachtungen insgesamt.



$$P_{GT}(unseen) = \frac{3}{18}$$

# Good-Turing-Smoothing:

- 10 red balls,
- 3 green balls,
- 2 blue balls
- 1 black ball
- 1 brown ball
- 1 grey ball

$$\sum_{x \in D} P(x) = 1,$$

wobei D der Bereich der möglichen Werte für x ist.

Good-Turing-Smoothing löst dieses Dilemma, indem es die Häufigkeitswerte anpasst. Es wird so getan, als ob Arten, die eigentlich r-mal vorkommen, r∗-mal vorkommen,mit

$$r* = \frac{(r+1)N_{r+1}}{N_r}. \qquad r* = \frac{(r+1)N_{r+1}}{N_r} = \frac{2 \cdot 1}{3}.$$

$$P_{GT}(x) = \frac{r*}{Z} \qquad P_{GT}(black) = \frac{2 \cdot \frac{1}{3}}{18} = \frac{2}{3 \cdot 18},$$

$$P_{GT}(unseen) = \frac{N(1)}{N} = \frac{3}{18} = \frac{1}{6}$$

# Katz Backoff Smoothing.

Ähnlich an Good Turing Berechnung.

$$P_{bo}(w_i \mid w_{i-n+1} \cdots w_{i-1}) = \begin{cases} d_{w_{i-n+1} \cdots w_i} \dfrac{C(w_{i-n+1} \cdots w_{i-1} w_i)}{C(w_{i-n+1} \cdots w_{i-1})} & \text{if } C(w_{i-n+1} \cdots w_i) > k \\ \alpha_{w_{i-n+1} \cdots w_{i-1}} P_{bo}(w_i \mid w_{i-n+2} \cdots w_{i-1}) & \text{otherwise} \end{cases}$$

*Katz, S. M. (1987). Estimation of probabilities from sparse data for the language model component of a speech recognizer. IEEE Transactions on Acoustics, Speech, and Signal Processing, 35(3), 400–401.*

# Wie evaluiert man ein N-Gramm Modell?:

Wir prüfen mit einem Testsatz ob unser Modell diesen gut vorhersagen kann

Wichtigstes Maß: Perplexität

$$Perplexity(W) = (\prod_{i=1}^{N} P(w_i|w_1, w_2, ..., w_{i-1}))^{-\frac{1}{N}}$$

Diese Form bringt aber einige Probleme mit sich!

*[1] https://huggingface.co/docs/transformers/en/perplexity*
*[2] https://latex.codecogs.com/eqneditor/editor.php (Only graphics)*
*[3] https://ar5iv.labs.arxiv.org/html/2210.05892*

# Wie evaluiert man ein N-Gramm Modell?:

Deshalb nutzen wir:

$$Perplexity(W) = e^{-\frac{1}{N}\sum_{i=1}^{N} lnP(w_i|w_1,w_2,...,w_{i-1})}$$

**Weil**:

$$ln(a * b) = ln(a) + ln(b)$$

$$e^{ln(x)} = x$$

$$(e^a)^b = e^{a*b}$$


*[1] https://huggingface.co/docs/transformers/en/perplexity*
*[2] https://latex.codecogs.com/eqneditor/editor.php (Only graphics)*
*[3] https://ar5iv.labs.arxiv.org/html/2210.05892*

# Programmvorstellung

# Wie funktioniert das Modell:

| | Das | ist | ein | Beispieltext |
|---|---|---|---|---|
| Das | 0 | 1 | 0 | 0 |
| ist | 0 | 0 | 1 | 0 |
| ein | 0 | 0 | 0 | 1 |
| Beispieltext | 0 | 0 | 0 | 0 |
| (Das ist) | 0 | 0 | 1 | 0 |
| (ist ein) | 0 | 0 | 0 | 1 |
| (Das ist ein) | 0 | 0 | 0 | 1 |

Model[Zeile][Spalte]
Model[Prefix][Sufix]

Danach Umrechnung in bedingte Wahrscheinlichkeit

# Zukunft von N-Gramm Modellen:

Werden vor allem durch Transformermodelle ersetzt.

**Transformermodelle**

| Vorteil | Nachteil |
|---|---|
| Versteht Kontext von (langen) Texten | Hoher Rechenaufwand/Speicherbedarf |
| Sehr hohe Genauigkeit | Schwierige Implementierung |
| Flexibel (Kann mit Rechtschreibfehlern umgehen) | Hohe Kosten für Entwicklung |
| Zugänglichkeit durch API‘s | Hoher Aufwand für das Training |

**N-Gramm Modelle**

| Vorteil | Nachteil |
|---|---|
| Einfache Implementierung | Kein bis „wenig“ Kontext |
| Geringe Kosten | Schlecht bei unseen ngrams/Rechtschreibfehlern |
| Ressourcenschonend | Große Datenmenge, um brauchbar zu werden |

*https://link.springer.com/chapter/10.1007/978-981-99-1999-4_2*
*https://iq.opengenus.org/n-gram-language-model-in-nlp/*
*https://www.kdnuggets.com/2022/06/ngram-language-modeling-natural-language-processing.html*

# Literatur:


- Daniel Jurafsky and James H. Martin, Speech and Language Processing, 3rd Edi-tion, Pearson, 2023.
- Dremio, “Data Sparsity,” https://www.dremio.com/wiki/data-sparsity/, Ac-cessed: June 1, 2024.
- David Foster, Generative Deep Learning: Teaching Machines to Paint, Write,Compose, and Play, O’Reilly Media, 2020.
- Prof. Dr. Johannes Maucher https://griesshaber.pages.mi.hdm-stuttgart.de/nlp/04ngram/01ngram.html, Accessed: June 1, 2024.
- William A. Gale, ”Good–Turing smoothing without tears,” Journal of Quantita-tive Linguistics, 1995.
- Roshmita Dey , Accessed: June 1,Understanding Language Modeling: From N-grams to Transformer-based Neural Models 2024.
- Alexander Clark, Gianluca Giorgolo, and Shalom Lappin, Statistical Representa-tion of Grammaticality Judgements: the Limits of N-Gram Models, Departmentof Philosophy, King’s College London
- A. Suresh Babu, Kumar P.N.V.S.Pavan Comparing Neural Network Approach withN-Gram Approach for Text Categorization, January 2010International Journal onComputer Science and Engineering

# Links:

- Wie funktioniert Sprache:
  - sketchengine.eu
- Literatur erfoschen:
  - fortext.net
  - books.google.com/ngrams/
- Sprachmodelle mit Python:
  - Statistische Sprachmodelle: nltk.org
  - Transformer Modelle/ Neuronale Netzwerke: huggingface.co
- Computerlinguistik:
  - cl.uni-heidelberg.de/
- Videoreihe zum Thema auf YT:
  - Kanal: From Language to Information

# Fragen:

- Fall: niedrige Wahrscheinlichkeit (richtige) gegen hohe Wahrscheinlichkeit
  - Wie gehen wir mit dem Fall um, wenn es eine sehr niedrige Wahrscheinlichkeit die richtige Wortwahl ist, diese aber gegen eine hohe Wahrscheinlichkeit abzuwiegen ist?

- Fall: Altdeutsch vs. Neudeutsch
  - Kann ich in alltäglicher Sprache mit einem Modell sprechen, dass nur auf Goethe und Schiller texten basiert?