Computer-Linguistische Anwendungen

CLA | B.Sc. | LMU

SVD Diskussion

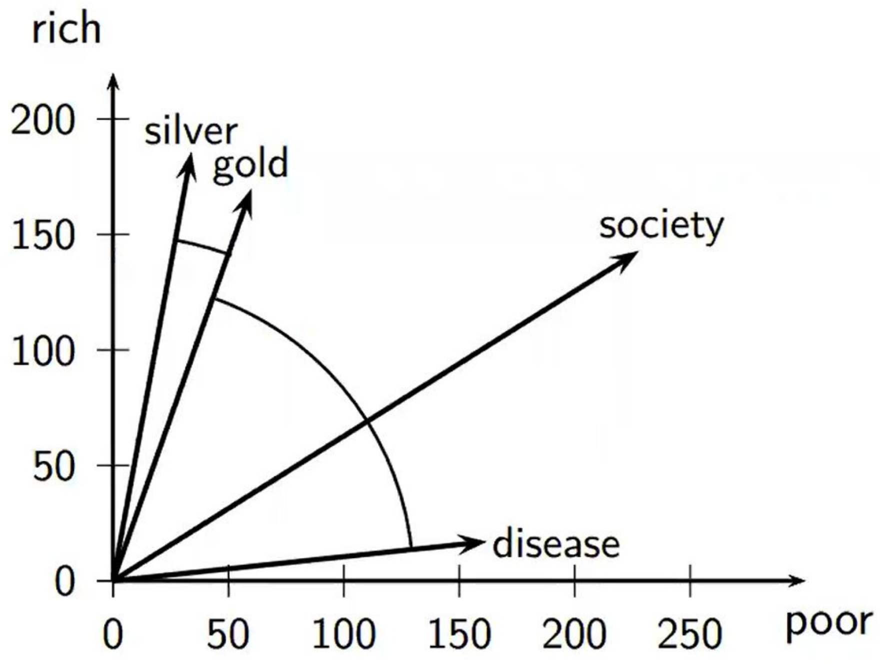
- SVD ist optimal:
  - Behält man die *k* größten Singulärwerte und setzt alle anderen auf 0, dann erhält man die optimale Annäherung der ursprünglichen Matrix C (Eckart-Young Theorem).

- Optimal: Keine andere Matrix desselben Rangs (= mit derselben unterliegenden Dimensionalität) ist eine bessere Annäherung an C.

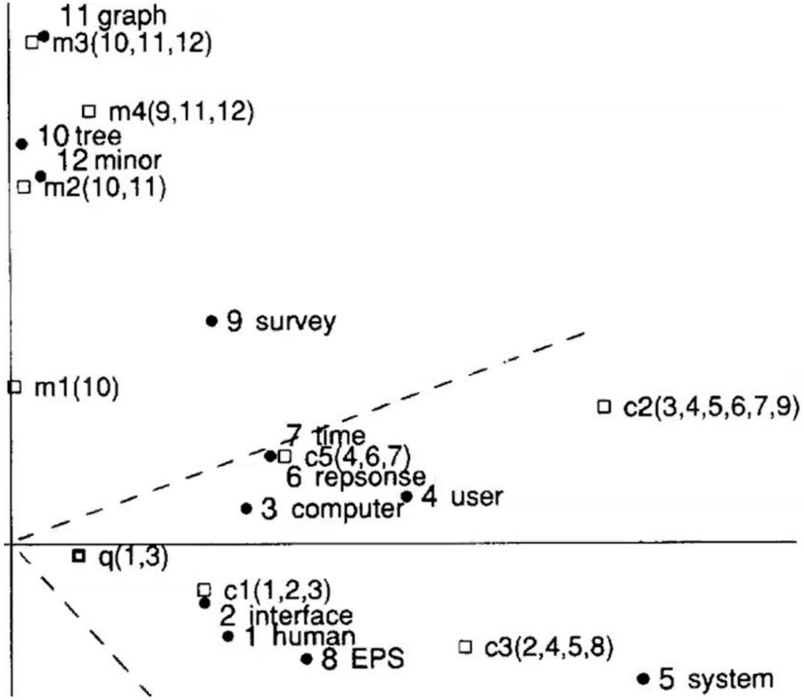
- Das Maß dieser Annäherung wird mit der Frobenius Norm bestimmt: $||C - C'||_F = \sqrt{\sum_i \sum_j (c_{ij} - c'_{ij})^2}$

- SVD verwendet demnach die "bestmögliche" Matrix, von der es demnach auch nur eine gibt.

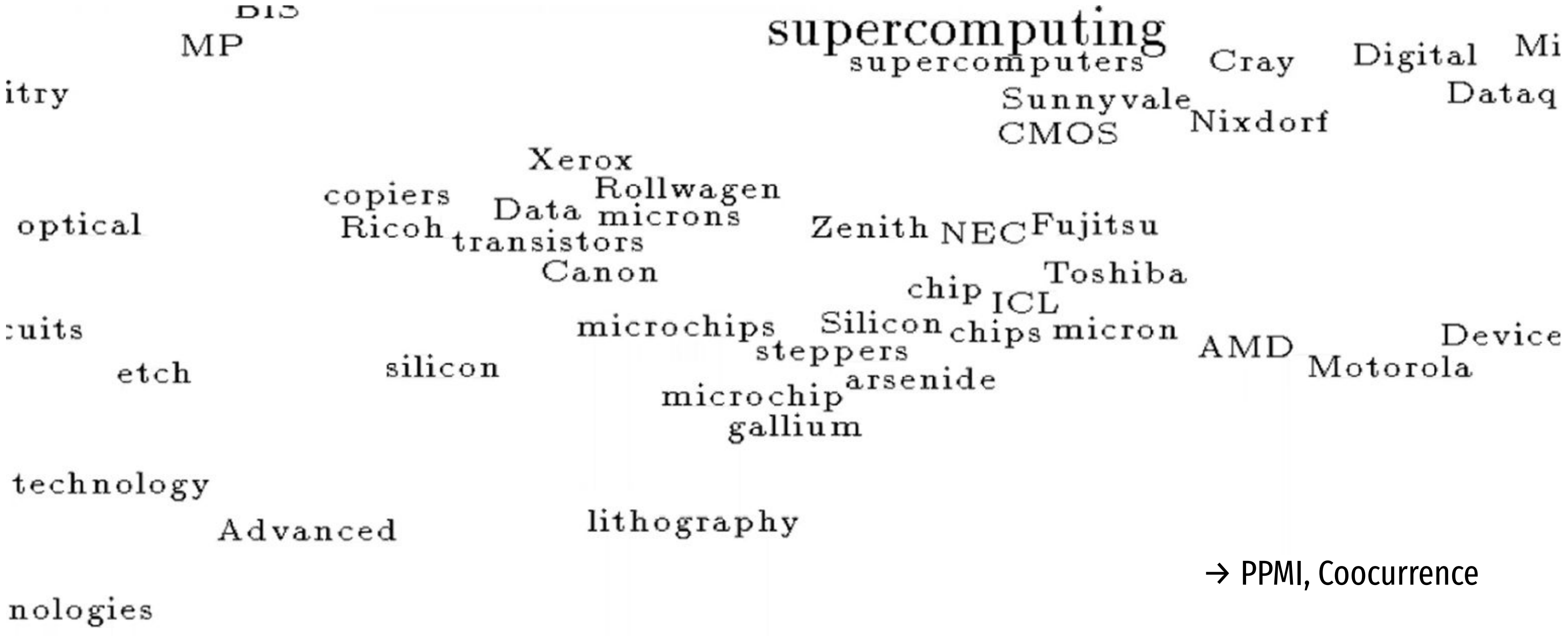
Embeddings (1): Vektor Raum Model (Salton, 1960s)



Embeddings (2): Latent Semantic Indexing (Deerwester, Dumais, Landauer, ..., 1980s)

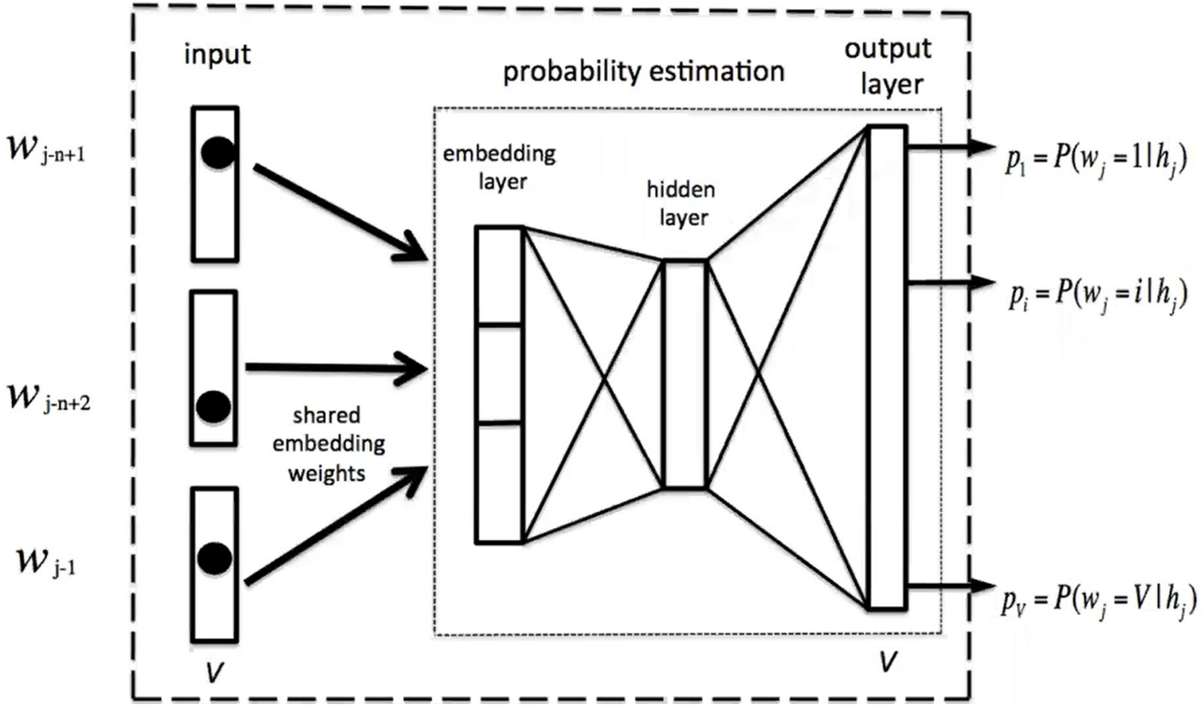


Embeddings (3): SVD-basierte Methoden (Schütze, 1992)

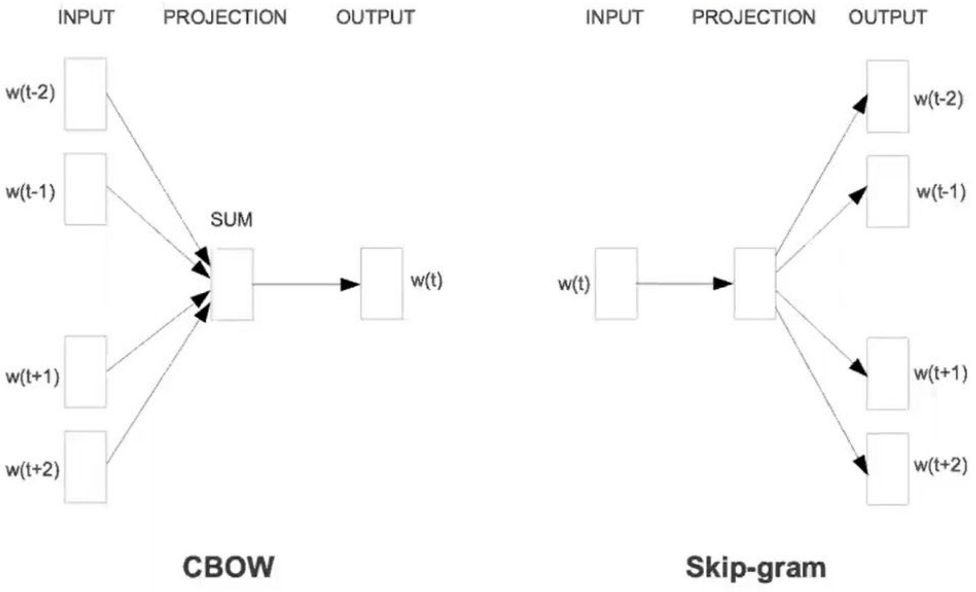


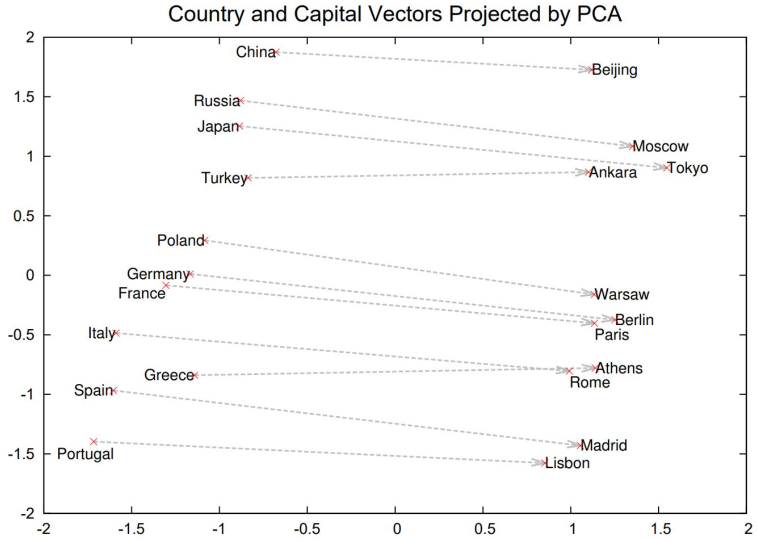
→ PPMI, Coocurrence

Embeddings (4): Neural models (Bengio, Schwenk, ..., 2000s)



Embeddings (5): Word2Vec (Mikolov, 2013)





Embeddings (6): SVD-based methods (Stratos et al., 2015)

SPECTRAL-TEMPLATE
Input: word-context co-occurrence counts $\#(w, c)$, dimension m, transformation method t, scaling method s, context smoothing exponent $\alpha \leq 1$, singular value exponent $\beta \leq 1$
Output: vector $v(w) \in \mathbb{R}^m$ for each word $w \in [n]$
Definitions: $\#(w) := \sum_c \#(w, c)$, $\#(c) := \sum_w \#(w, c)$, $N(\alpha) := \sum_c \#(c)^\alpha$

1. Transform all $\#(w, c)$, $\#(w)$, and $\#(c)$:

$$\#(\cdot) \leftarrow \begin{cases} \#(\cdot) & \text{if } t = \text{—} \\ \log(1 + \#(\cdot)) & \text{if } t = \log \\ \#(\cdot)^{2/3} & \text{if } t = \text{two-thirds} \\ \sqrt{\#(\cdot)} & \text{if } t = \text{sqrt} \end{cases}$$

2. Scale statistics to construct a matrix $\Omega \in \mathbb{R}^{n \times n}$:

$$\Omega_{w,c} \leftarrow \begin{cases} \#(w, c) & \text{if } s = \text{—} \\ \frac{\#(w,c)}{\#(w)} & \text{if } s = \text{reg} \\ \max\left(\log \frac{\#(w,c)N(\alpha)}{\#(w)\#(c)^\alpha}, 0\right) & \text{if } s = \text{ppmi} \\ \frac{\#(w,c)}{\sqrt{\#(w)\#(c)^\alpha}} \sqrt{\frac{N(\alpha)}{N(1)}} & \text{if } s = \text{cca} \end{cases}$$

Perform rank-m SVD on $\Omega \approx U \Sigma V^\top$ where $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_m)$ is a diagonal matrix of ordered singular values $\sigma_1 \geq \cdots \geq \sigma_m \geq 0$.

Define $v(w) \in \mathbb{R}^m$ to be the w-th row of $U\Sigma^\beta$ normalized to have unit 2-norm.

Embeddings (7): GloVe (Pennington, Socher, Manning, 2014)

$$J = \sum_{i,j=1}^{V} f(X_{ij})(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij})^2$$

Takeaway: Limitations of WordSpace

- WordSpace Vektoren können ineffizient sein
(große Zahl an Parametern, wenn es für maschinelles (deep) learning verwendet wird

- WordSpace Vektoren können ineffizient sein
(wegen Zufall und Noise in den Coocurrences)

Takeaway: Definition des Embedding

- Realwert Vektor Repräsentation eines Wortes *w*
- Repräsentiert semantische und andere Eigenschaften von *w*
- Niedrige Dimensionalität *k* (e.g., $50 \leq k \leq 1000$)
- Dicht (dense, im Gegensatz zu sparse)

Takeaway: Embedding bei Matrix Faktorisierung

- Berechnung der PPMI Coocurrence Matrix
- Singulärwertzerlegung (SVD)
- Reduktion der linken Matrix U in d Dimensionen
- Reduktion von U ist dann die Embedding Matrix

Weitere Informationen

- Kapitel 18 des IIR auf http://cislmu.org
- Deerwester et al.'s paper über Latent Semantic Indexing
- Paper über probabilistic LSI von Thomas Hofmann
- "Neural Network Embeddings as Implicit Matrix Factorization" (Levy, Goldberg, 2014)