

Quantitative Textanalyse

Sitzung 12: Embedding Regression

Mirko Wegemann

08. Januar 2025

Unsere heutige Sitzung

- Von bags-of-words zu einer komplexeren Repräsentation von Text
- Wir verwenden Embeddings, um den Kontext zu verstehen, in dem bestimmte Begriffe verwendet werden
- Mithilfe von Embedding Regression sehen wir, wie sich die Feature-Nutzung je nach Kontext verändern kann

Warum bags-of-words approaches nicht immer ausreichen...

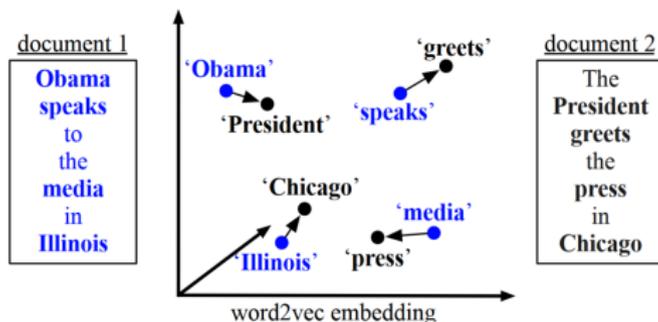
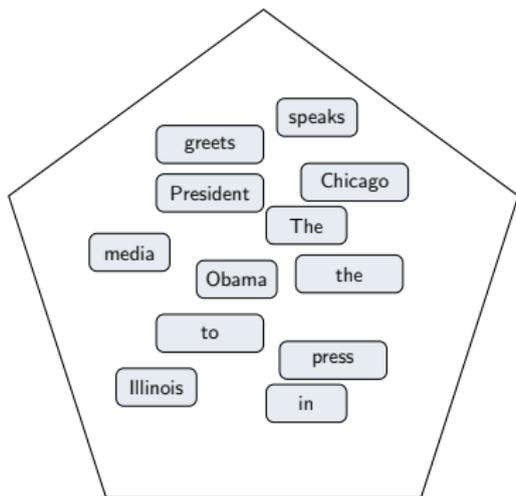


Figure 1. An illustration of the *word mover's distance*. All non-stop words (**bold**) of both documents are embedded into a *word2vec* space. The distance between the two documents is the minimum cumulative distance that all words in document 1 need to travel to exactly match document 2. (Best viewed in color.)

Kusner et al. (2015)

Was bags-of-words approaches tun würden... I



Was bags-of-words approaches tun würden... II

```
> dfm_example
Document-feature matrix of: 2 documents, 12 features (37.50% sparse) and 0 docvars.
  features
docs  obama speaks to the media in illinois . president greets press chicago
text1  1     1  1  1  1     1  1     1  1     0     0     0     0
text2  0     0  0  2  0  1     0  1     1     1     1     1
```

So sieht unser Text in einer bags-of-words-Struktur aus

Was könnte hier problematisch sein?

Was bags-of-words approaches tun würden... III

Der größte Nachteil von bags-of-words-Ansätzen ist, dass sie weder die (1) **Beziehung eines Wortes zu anderen Wörtern** noch die (2) **Position eines Wortes** innerhalb eines Satzes erfassen.

Sie sind kontextblind.

Was bags-of-words approaches tun würden... IV

Wir können zwischen zwei Arten von Embeddings unterscheiden:

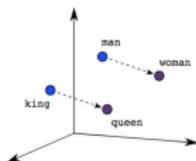
1. statische Embeddings, wie GloVe (Pennington et al. 2014)
2. kontextuelle Embeddings, wie BERT (Peters et al. 2018)

Die Idee hinter Word Embeddings I

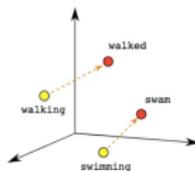
- Idee: Ähnlichkeit von Wörtern erfassen
- wir können einem Wort einen **mehrdimensionalen Vektor** zuweisen, der seine Beziehung zu anderen Wörtern darstellt
- “distances between such vectors are informative about the **semantic similarity** of the underlying concepts they connote for the corpus on which they were built” (P. Rodriguez and Spirling 2021)
- Word Embeddings “predict the occurrence of a word by the surrounding word in a text sequence” (Rheault and Cochrane 2020, p. 112)

Die Idee hinter Word Embeddings II

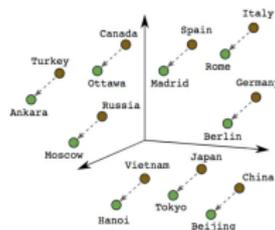
Embeddings transformieren Features in einen k-dimensionalen Raum (hier simplifiziert in drei Dimensionen).



Male-Female



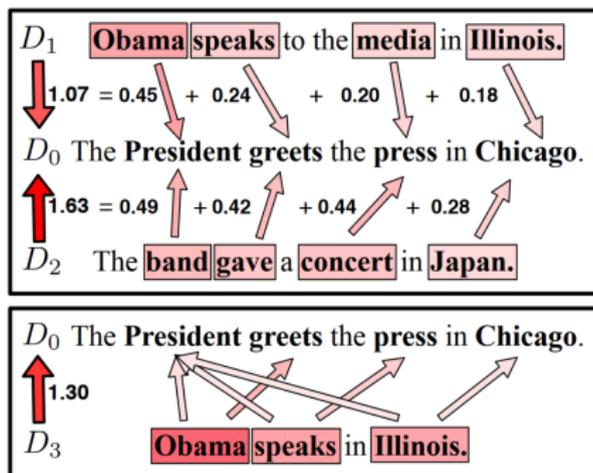
Verb Tense



Country-Capital

1

Die Idee hinter Word Embeddings III



Kusner et al. (2015)

¹Quelle:

<https://towardsdatascience.com/a-guide-to-word-embeddings-8a23817ab60f>

Wofür Embeddings nützlich sein können...

P. Rodriguez and Spirling (2021) unterscheiden zwischen zwei Funktionen:

1. Embeddings haben einen **intrinsischen** Wert (Analyse auf der Wortebene)
 - zum Beispiel: wenn der Abstand zwischen dem Begriff 'Migrant' und 'fleißig' für Grüne näher ist als für Konservative, könnten wir daraus Muster von Parteikommunikation ableiten
2. ...aber auch einen **instrumentellen** Wert: sie erfassen mehr Bedeutung als bags-of-words → sie könnten unsere *classification tasks* verbessern

Entscheidungen bei der Nutzung von Embeddings I

- **word2vec** vs. **GloVe** (Unterschied im Training von Embeddings)?
- **pre-trained** vs. **locally-trained** Embeddings?
- **Kontextfenster** (window size): Wie viel Kontext wird berücksichtigt?
- **Dimensionalität**: Wie viele Dimensionen berücksichtigen wir?

Entscheidungen bei der Nutzung von Embeddings II

pre-trained vs. local

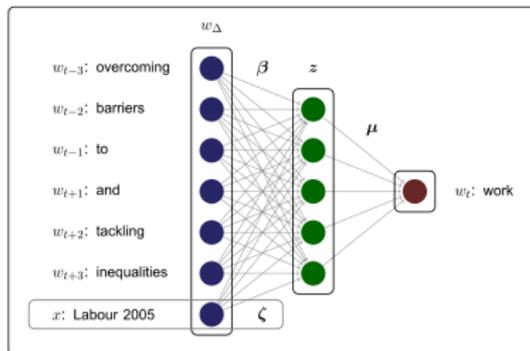
- Repräsentation eines Features wird normalerweise durch deep learning-Ansätze (wie neuronale Netze) gelernt
- Wir können entweder korpus-spezifische Embeddings erstellen, die die domain-spezifische Beziehungen lernen
- ...oder pre-trained Embeddings verwenden (, welche mithilfe eines großen Korpus wie der Wikipedia vorab trainiert worden sind)
- Entscheidung hängt von Korpus-Größe und Spezifität ab

Entscheidungen bei der Nutzung von Embeddings III

Kontextfenster

- Je größer das Kontextfenster (*window size*), desto weiter entfernte Wörter beeinflussen den Embedding-Vektor

Wir sagen ein Wort w_t durch die umgebende Wörter w_{t-1} , w_{t-2} , ..., w_{t-n} vorher; $\rightarrow n$ ist dabei das Kontextfenster



Rheault and Cochrane (2020, p. 116)

Entscheidungen bei der Nutzung von Embeddings IV

Dimensionalität

- Die Anzahl der Dimensionen kann frei gewählt werden
 - Der Begriff *Dimensionen* bezieht sich auf die Länge der Vektor-Darstellung eines Features
 - Je mehr Dimensionen, desto mehr Merkmale eines Wortes erfassen wir, aber desto rechenintensiver wird es (und desto eher neigen Modelle dazu, zu overfitten)

Entscheidungen bei der Nutzung von Embeddings V

In der Praxis hat die Wahl dieser *Hyperparameter* nur marginale Auswirkungen – pre-trained Modelle funktionieren gut (P. Rodriguez and Spirling 2021)

Verwendung von Word Embeddings I

Es gibt drei verschiedene Methoden, um Embeddings zu erhalten

1. Trainieren von eigenen "lokalen" Embeddings mit einem neuronalen Netzwerk
2. Nutzung vorab trainierter Embeddings (z.B. für GloVe oder andere mehrsprachige Embeddings)
3. fine-tuning von vorab trainierten Embeddings

Verwendung von Word Embeddings II

Je nach Methode sind unterschiedliche Schritte erforderlich. Für **pre-trained embeddings**:

- Datenvorbereitung
- Übereinstimmung zwischen Input und Embeddings
- Datenanalyse

Für **locally trained embeddings**:

- Datenvorbereitung
- Neuronales Netzwerk, um die Embeddings zu lernen
- Datenanalyse

Wir konzentrieren uns hier auf pre-trained Embeddings (die wir später fine-tunen), aber es gibt Code am Ende des Skripts zum Trainieren eigener Embeddings.

Vorbereitung in R

Wir müssen nicht unbedingt pre-processing betreiben, aber oftmals nehmen wir folgende Schritte vor:

- Entfernen häufiger Wörter (oder stop words) ohne viel semantische Bedeutung kann unsere Embeddings verbessern
- Entfernen von Punctuation und Ziffern
- Erstellung von n-grams (z.B. Europäische_Union) oder Lemmatisierung kann zusätzlich Interpretation erleichtern

...und nun in R

Deskriptive Analyse I

nearest neighbors (welche Wörter sind nah beieinander?) Ein oft genutztes Beispiel (s. zuvor) ist die Gleichung

$$\textit{Berlin} = \textit{Paris} - \textit{Frankreich} + \textit{Deutschland} \quad (1)$$

Nicht nur geographisch, sondern auch semantisch sollte der Abstand zwischen dem Wort Berlin und Deutschland gleich dem zwischen Paris und Frankreich sein.

Was ist die Hauptstadt von Deutschland?

Wir können diese Gleichung in R übersetzen.

```
1 > which_capital = embeddings["paris", , drop = FALSE] -  
2 + embeddings["france", , drop = FALSE] +  
3 + embeddings["germany", , drop = FALSE]  
4 > capital_cos_sim = sim2(x = embeddings, y =  
5   which_capital, method = "cosine", norm = "l2")  
6 > head(sort(capital_cos_sim[,1], decreasing = TRUE), 5)  
7 berlin    germany frankfurt  hamburg    paris  
0.7635345  0.7232592  0.6718858  0.6530555  0.6509711
```

Weitere deskriptive Analysen

Begriffe in der Nähe von "migrant"

```
1 > find_nns(embeddings['migrant',], pre_trained =
2   embeddings, N = 20)
3 [1] "migrant"      "immigrant"    "refugee"
4   "worker"      "undocumented" "farmworker"
5   "indigenous"
6 [8] "unskilled"    "migration"    "expatriate"
7   "immigration" "migratory"    "resettlement" "labor"
8 [15] "employment"  "unemployed"  "population"
9   "plight"      "welfare"     "labour"
```

Deskriptive Analysen II

Wir können mithilfe des `conText`-Packages auch Ähnlichkeiten von Features je nach Kovariaten anschauen (welche Wörter werden von welchen Akteur*innen im Kontext von Wort w genutzt?)

```
1 context_wv_parfam <- dem_group(context_dem, groups =  
  context_dem@docvars$parfam)  
2 dim(context_wv_parfam)  
3  
4 context_nns <- nns(context_wv_parfam, pre_trained =  
  embeddings, N = 10, candidates =  
  context_wv_parfam@features, as_list = TRUE)  
5 context_nns
```

...und nun in \mathbb{R}

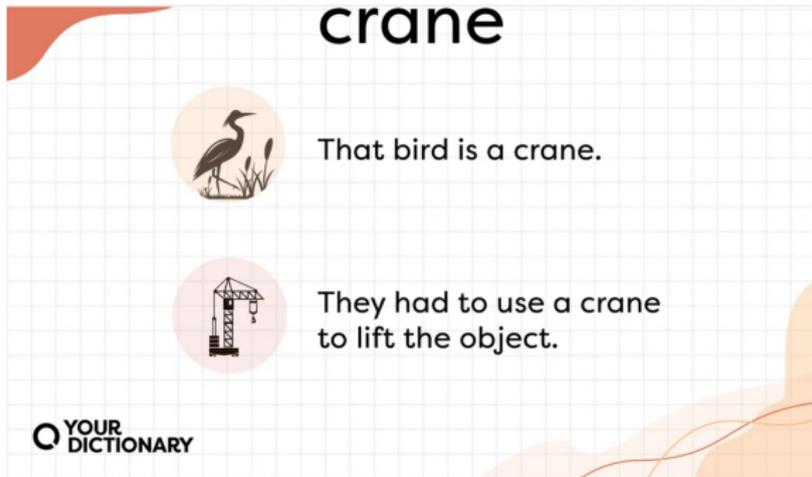
Embedding Regression I

- **kontextuelle Verwendung** eines Begriffs (wir kommen kontextuellen Embeddings näher)
- hierfür nutzen wir Kovariaten (wie zuvor in der deskriptiven Analyse)
- wir **mitteln quasi ein Embedding** in unterschiedlichen Kontexten und vergleichen, wie ähnlich es ist (zuvor gewichten wir Wörter, die häufig in beiden Kontexten vorkommen, als weniger relevant)

P. L. Rodriguez et al. (2023)

Embedding Regression II

Da wir den Kontext berücksichtigen, können wir nun unterschiedliche Wortbedeutungen erfassen:



crane

 That bird is a crane.

 They had to use a crane to lift the object.



Embedding Regression in R

conText ermöglicht es, Ähnlichkeiten nach Gruppen für jeden Zielbegriff zu schätzen und liefert darüber hinaus Standardfehler

```
1  set.seed(451)
2  model2 <- conText(formula = trump ~ prepost,
3  data = toks,
4  pre_trained = embeddings,
5  transform = TRUE, transform_matrix = trans_mat,
6  bootstrap = TRUE,
7  num_bootstraps = 100,
8  permute = TRUE, num_permutations = 10,
9  window = 10,
10 verbose = T)
```

...und nun in \mathbb{R}

Evaluation von Embeddings I

- für Downstream-Aufgaben (z. B. Klassifikation mit Embeddings) können wir konventionelle Metriken des maschinellen Lernens verwenden (s. Sitzung 10: F1-Score, Accuracy, confusion matrix, etc.)
- für intrinsische Aufgaben schwieriger: Was macht ein Embedding besonders gut?

Evaluation von Embeddings II

P. Rodriguez and Spirling (2021) schlagen die 'Turing-Validierung' vor

- Computer performen dann gut, wenn sie nicht von menschlichen Aufgaben zu unterscheiden sind
 - in der Praxis: Menschen vergleichen eine Liste von Wortassoziationen, die von Menschen erstellt wurde, mit einer Liste von Embeddings
- wenn Menschen beide Listen nicht unterscheiden können, ist die Leistung gut
- andere Validierungen umfassen die Korrelation zwischen Modellen

Evaluation von Embeddings III

- bei *scaling tasks* wie dem von Rheault and Cochrane (2020):
cross validation mit anderen Maßen zu Parteipositionen

Was wir heute gelernt haben...

- ...was Embeddings sind und was sie von bags-of-words Ansätzen unterscheidet
- ...wie wir sie für unsere Forschung nutzen können
- ...wie Embeddings Regression uns hilft, zusätzlichen Kontext in der Nutzung von Wörtern einzufangen

In der nächsten Woche...

- ...haben wir unsere letzte inhaltliche Sitzung
- ...beschäftigen wir uns mit dem instrumentellen Zweck von Word Embeddings und bauen ein neural network zur Klassifikation von Text
- ...wenn ihr der Sitzung folgen wollt, empfiehlt es sich Python zu installieren und vor der Sitzung die Präambel des R-Skripts durchlaufen zu lassen

Literatur I

- Kusner, M. J., Sun, Y., Kolkin, N. I., & Weinberger, K. Q. (2015). From Word Embeddings To Document Distances.
- Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 1532–1543.
- Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep contextualized word representations. <https://arxiv.org/abs/1802.05365>
- Rheault, L., & Cochrane, C. (2020). Word Embeddings for the Analysis of Ideological Placement in Parliamentary Corpora. *Political Analysis*, 28(1), 112–133. <https://doi.org/10.1017/pan.2019.26>

Literatur II

Rodriguez, P., & Spirling, A. (2021). Word Embeddings: What works, what doesn't, and how to tell the difference for applied research. *The Journal of Politics*.

<https://doi.org/10.1086/715162>

Rodriguez, P. L., Spirling, A., & Stewart, B. M. (2023). Embedding Regression: Models for Context-Specific Description and Inference. *American Political Science Review*, 117(4), 1255–1274. <https://doi.org/10.1017/S0003055422001228>