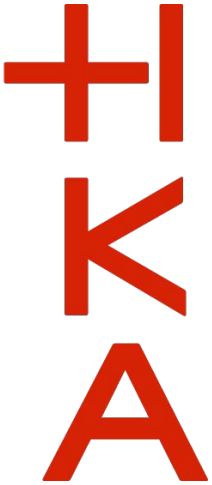
inovex



KI Labor - Wintersemester 2021

NLP Einführung

Darjan Salaj, Sven Müller, **Maximilian Blanck,**
Sebastian Blank, Frederik Martin, **Pascal Fecht**

Karlsruhe, 05.Nov. 2021

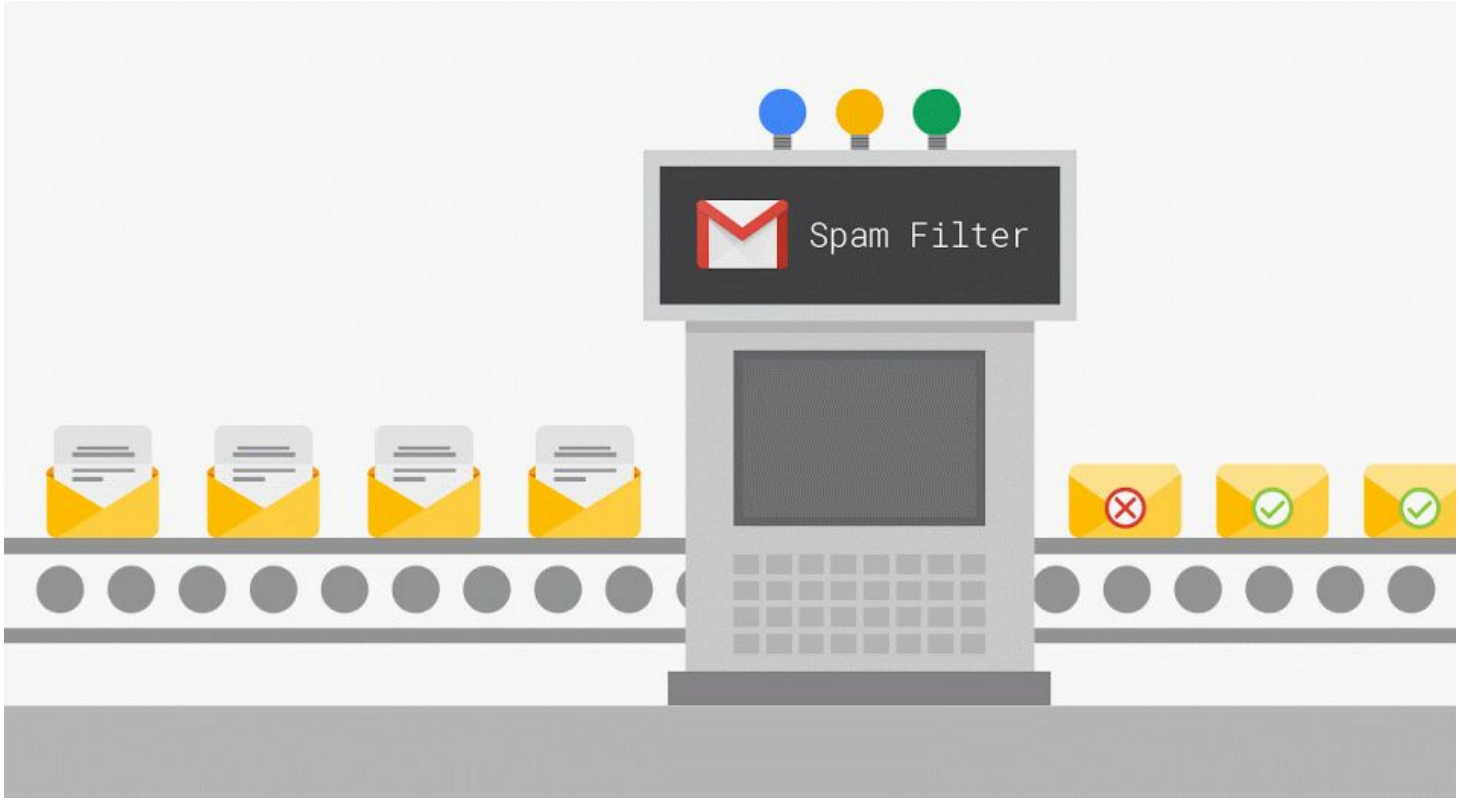
Schedule

Datum	Thema	Inhalt	Präsenz
01.10.21	**Allg.**	**Organisation, Teamfindung**	Nein
08.10.21	CV	Vorstellung CV	Nein
15.10.21	CV	Q&A Sessions	Nein
22.10.21	CV	Sprintwechsel, Vorstellung Assignment	Ja
29.10.21	CV	Q&A Sessions	Nein
05.11.21	**CV / NLP**	**Abgabe CV, Vorstellung NLP**	**Ja**
12.11.21	NLP	Q&A Sessions	Nein
19.11.21	NLP	Sprintwechsel, Vorstellung Assignment	Ja
26.11.21	NLP	Q&A Sessions	Nein
03.12.21	NLP	Q&A Sessions	Nein
10.12.21	**NLP / RL**	**Abgabe NLP, Vorstellung RL**	**Ja**
17.12.21	RL	Q&A Sessions	Nein
14.01.22	RL	Sprintwechsel, Vorstellung Assignment	Ja
21.01.22	RL	Q&A Sessions	Nein
28.01.22	**RL**	**Abgabe RL, Abschluss KI Labor**	**Ja**

Agenda

- **Theorie**
  - Natural Language Processing
  - Vorverarbeitung
  - Word Embeddings (word2vec)
  - Text Klassifikation → Sentiment Analyse
- **Übungsaufgaben**
  - Word Embeddings Alice im Wunderland(Aufgabe 1)
  - Sentiment Analyse für Twitter Posts (Aufgabe 2)

Natural Language Processing



https://i.gifer.com/Ou1t.gif

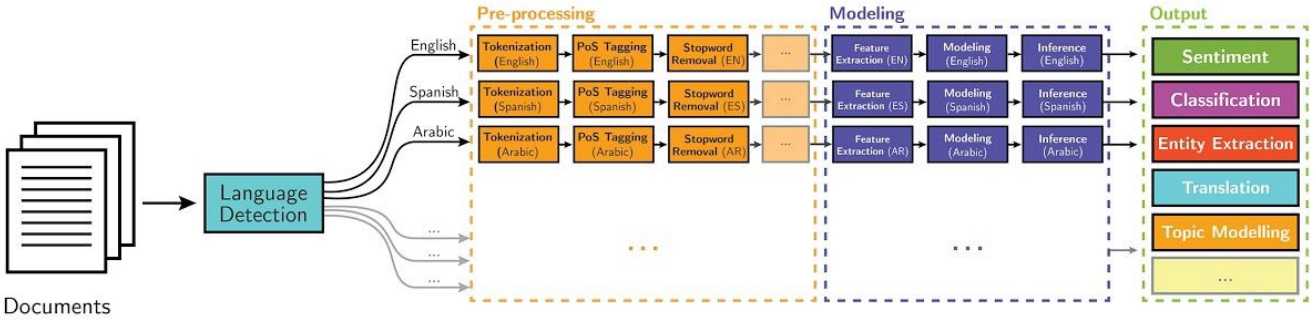
Natural Language Processing...

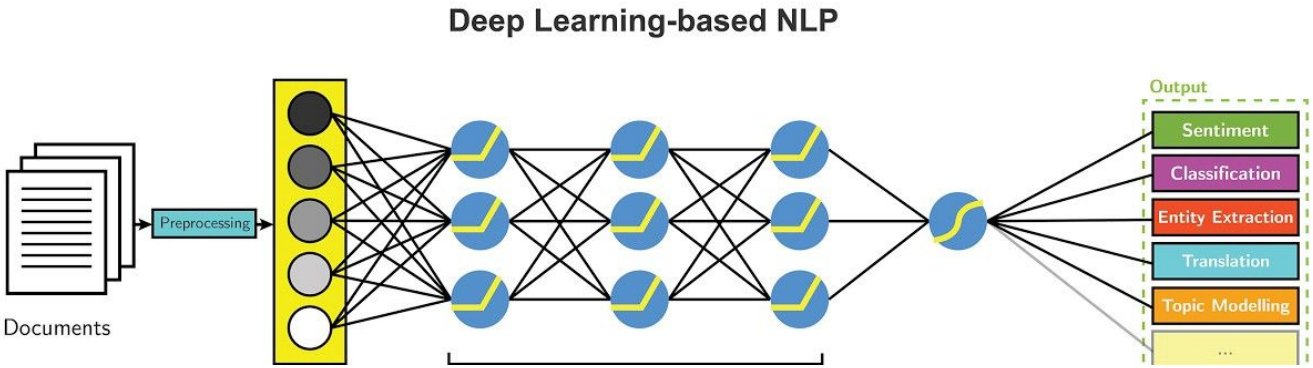
- **beinhaltet** Linguistik, Natürliche Sprache, Künstliche Intelligenz, Mathematik, …
- **befasst** sich mit der Interaktion zwischen Computern und Menschen
- **beschäftigt** sich unter anderem mit:

  text classification, named entity recognition, machine translation, part of speech tagging, sentiment analysis, question answering, text summarization, text generation, speech recognition, speech to text, text to speech, chatbots, virtual assistants, …

  → https://paperswithcode.com/area/natural-language-processing

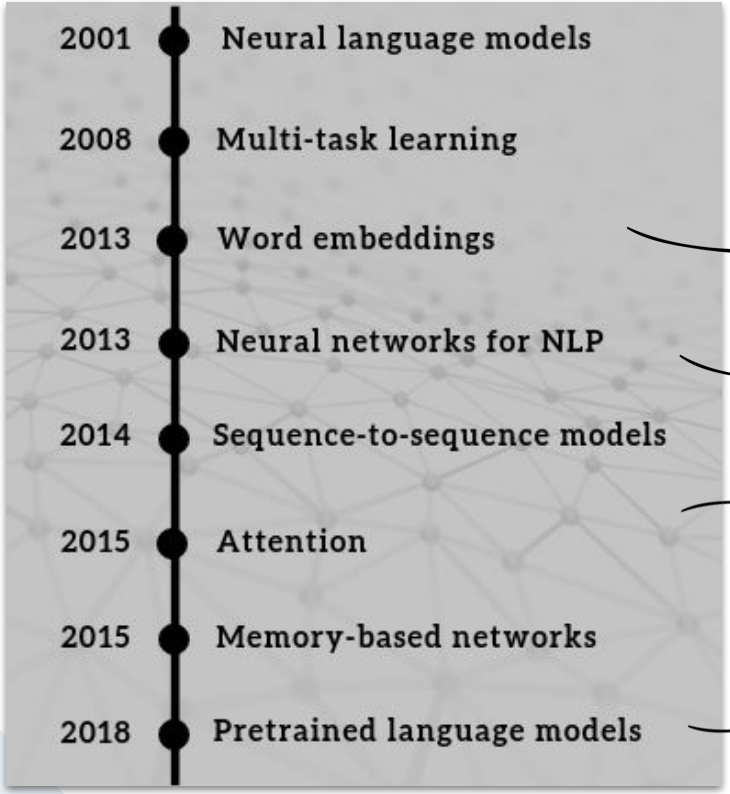
Klassische vs Deep Learning NLP Pipeline





https://towardsdatascience.com/introduction-to-natural-language-processing-for-text-df845750fb63

Die letzten 10 Jahre in NLP



1. Übungsaufgabe

2. Übungsaufgabe

3. Assignment

https://ruder.io/a-review-of-the-recent-history-of-nlp/

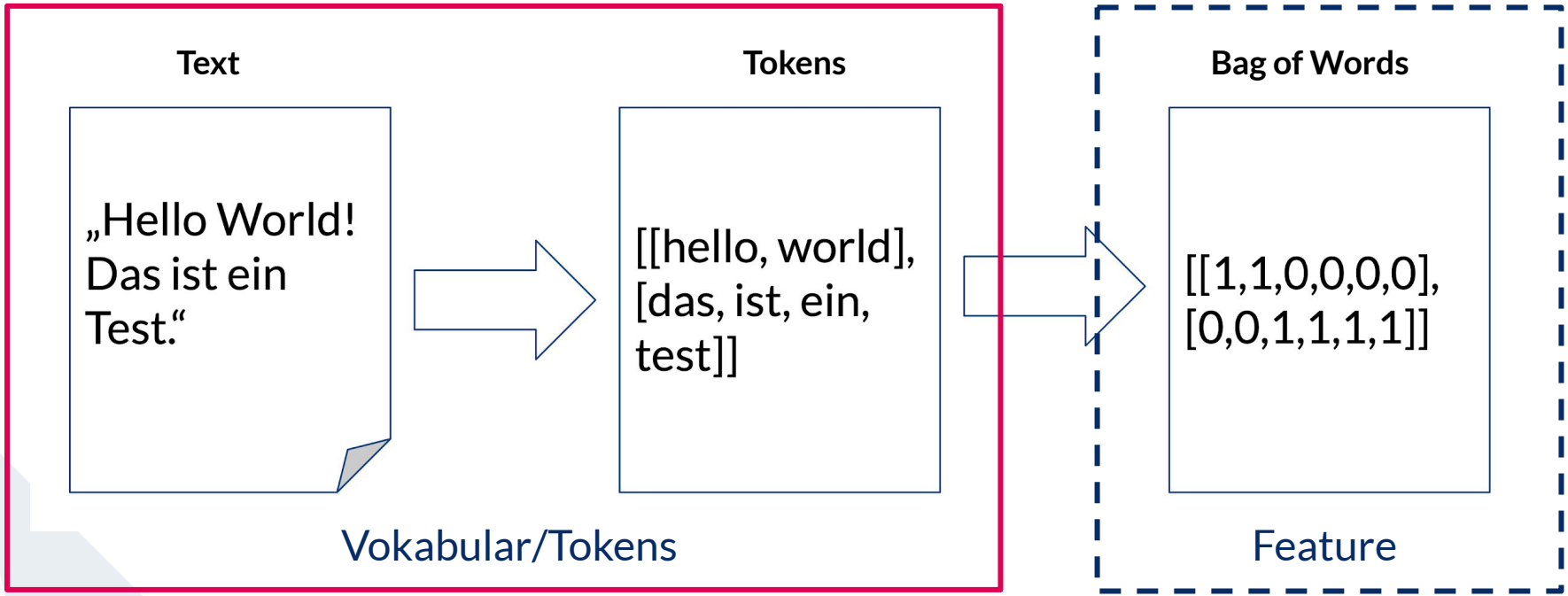
Herausforderungen in NLP

1. Sprache muss irgendwie in den Computer kommen.
   - Wort / Laut ⇒ Zahl(en) (Featurisierung)
2. Das Vokabular einer Sprache ist in der Regel sehr umfangreich
   - Dimensionalitätsproblem (?)
3. Einige Wörter sind enger verbunden als andere
   - Ähnlichkeit, Abstand ⇒ Maße?

Wie kommt Sprache in Computer?

Wie kommt die Sprache in den Computer?

Klassischer Ansatz: **Text → Tokens → Bag of Words**



https://machinelearningmastery.com/gentle-introduction-bag-words-model/

Vorverarbeitung
Vokabular & Features

Vorverarbeitung: Überblick

- **Vokabular**: Tokenization/Normalization:
  - Erkennung von Satz- / Wortgrenzen
  - Einschränkung des Vokabulars
  - Stop Word Removal
  - Lower Casing

- Umwandlung von Text in **Features**:
  - Part of Speech Tagging
  - Bag Of Words, **Embeddings, Context Embeddings**
  - Andere Features (Rechtschreibung, numerische Werte, Sprache, etc.)

Vokabular

Tokenisierung

Splitte Texte in kleinere Teile, Vor allem:

- “Wörter” → einfacher Ansatz: Trenne an “ “ (Leerzeichen):

```
re.compile(r'\W*\s+').split("Das Auto fährt")
```

→ [“Das”, Auto”, “fährt”]

- Aber: was ist mit “H-Milch”, “Auto-\nbahn“, “Dr. Musterdoktor”
- Sätze → Trenne an “.” Punkt?

Vokabular

Sind Wörter die richtige Granularität für Tokens?

Subwords

- Nicht im Vokabular enthaltene Token werden in Sub-Wörter geteilt.

Characters

- Jedes Char hat ein Encoding
- Nachteil: Die Bedeutung von Wörtern geht verloren.

Byte-Pair Encodings → Erklärung

- Mix aus Subword und Character Level Encoding

⇒ Wir fangen mit Wörtern, kommen aber darauf im Assignment zurück.

Vokabular

Der Eingabevektor von NLP Modellen

Grundfrage: Wie viele unterschiedliche Wörter soll mein ML Modell als Eingabe erhalten?

Typische Probleme:

- Sehr seltene Wörter gehen unter.
- Häufig Worte, die wenig Inhalt enthalten (“der”, “ein”, …) können starke Features werden.
- Wie gehen wir mit Noise um?

Vokabular

Einfaches Preprocessing

- **Restriktion des Vokabular**
  - Vokabluargröße von x Wörtern
  - Ein Wort muss mindestens y mal vorkommen, damit es “*relevant*” ist.
- **Lowercasing**

Vokabular

Einfaches Preprocessing

› Stoppwörter entfernen

```
1 from nltk.corpus import stopwords
2 stop_words = set(stopwords.words('english'))
3 tokens = [w for w in tokens if not w in stop_words]
```

› Und vieles mehr (Lemmatisierung, …)

NLTK

spaCy

https://www.nltk.org/, https://spacy.io/

Vokabular/Features

Tagging

Beispiel: Jede IBAN ist ein eigenes Token

⇒ Kein relevantes Feature für Modell

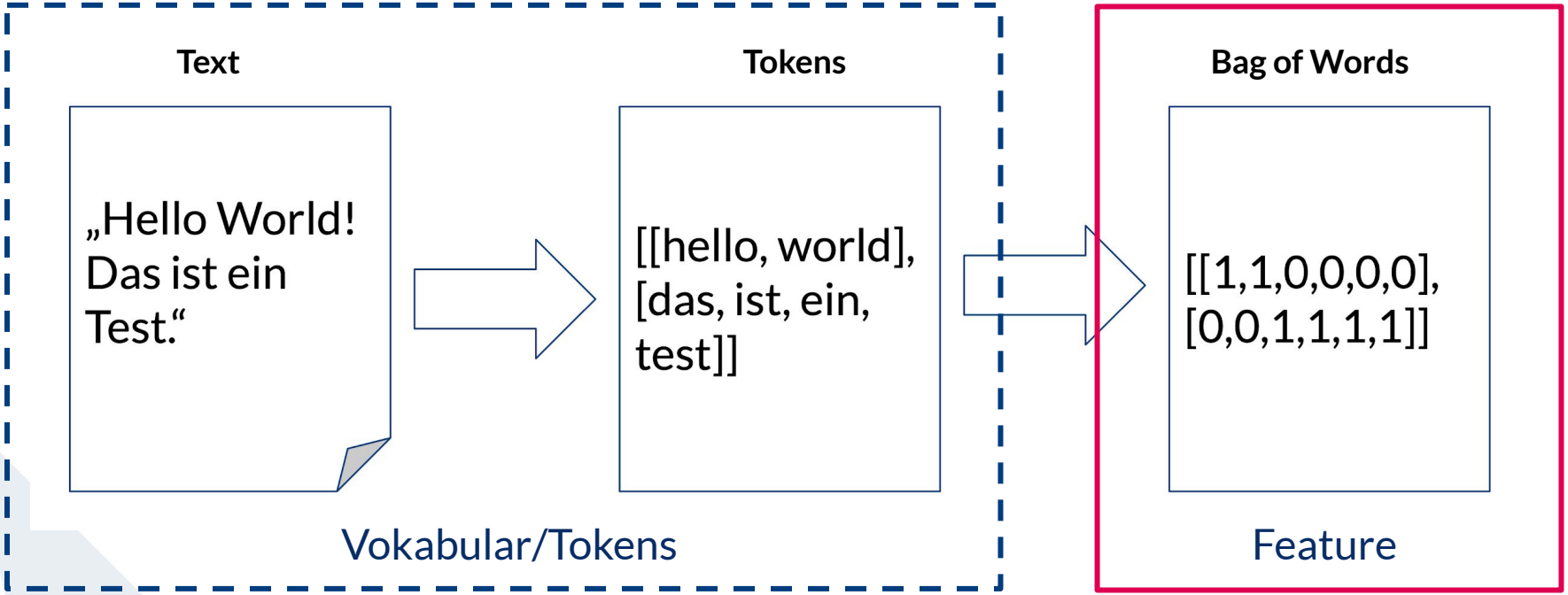
```
1 if re.match('^DE(?:\\s*[0-9a-zA-Z]\\s*){20}$', TOKEN):
2     return "<IBAN>"
```

⇒ Modell erkennt alle (deutschen) IBANs als ein einziges Feature.

Word Embeddings - Die **Bedeutung** von Wörtern

Wie kommt die Sprache in den Computer?

Klassischer Ansatz: **Text → Tokens → Bag of Words**



https://machinelearningmastery.com/gentle-introduction-bag-words-model/

Features

Bag of Words

Zähle alle unterschiedlichen Wörter in den Texten, die betrachtet werden:

- Baue einen Vektor mit Vokabularlänge
- Jedes Wort hat einen Index im Vektor
- Merke die Anzahl für jedes Wort und Speichere die Zahl im Vektor
- Feature für einen Text:
  - [0,0,2,1,0,0,...,0,0,1,3,0,0]

→ Problem: Der Vektor beinhaltet sehr viele “0”en

Word2Vec

- “You shall know a word by the company it keeps” (Firth, J. R. 1957:11)
- Word2Vec-Paper: https://arxiv.org/abs/1301.3781

Idee:

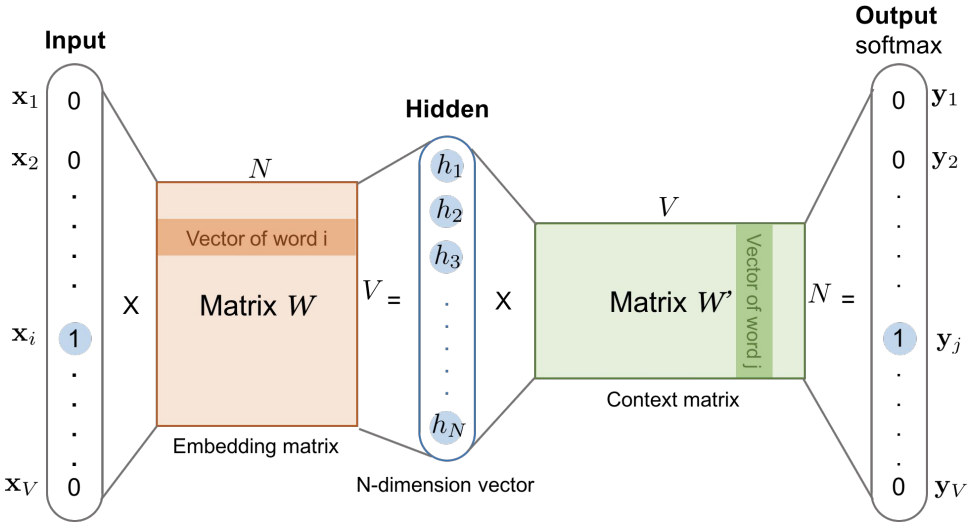
- Die umliegenden Wörter eines Wortes (in einem beliebigen Text) haben Einfluss auf die Bedeutung dieses Wortes.
- Beispielsatz: “*Die Ente* **quakt** *und tanzt*.”

Word2Vec

- Jedes Wort wird auf einen Vektor der Länge **n** abgebildet:
- Bsp.
  "Auto" → [0.012, 0.981, -0.271,...]
- **n** ist im Word2Vec-paper 300, andere Werte möglich
- Wie werden diese Werte erzeugt?
  - 2 Ansätze:
    - Skip-Gram
    - Cbow

Word2Vec - Skip-Gram (Kontext zum Wort)

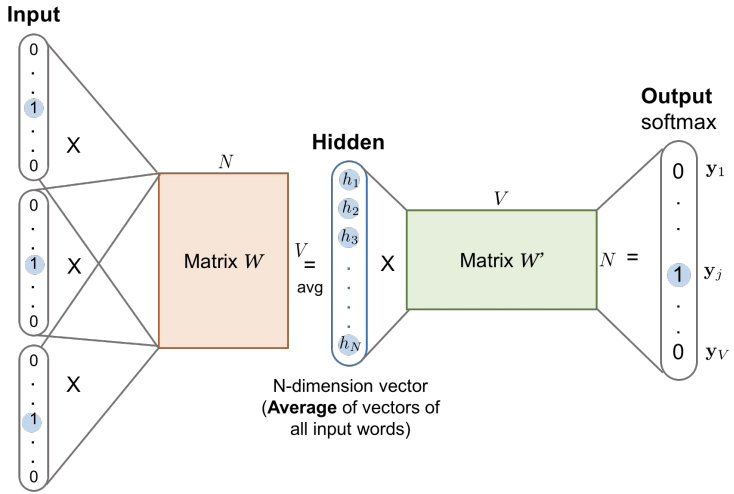
- Gegeben das Wort "**quakt**", wollen wir den Kontext vorhersagen
  - "*Die Ente* **quakt** *und tanzt.*" (Fenstergröße 5)
  - Kontext: ["Die", "Ente", "und", "tanzt"]; Target: ["**quakt**"]
  - Trainingsdaten: X_1=["**quakt**"], Y_1=["Die"]; X_2 =["**quakt**"], Y_2=["Ente"]; ...



https://lilianweng.github.io/lil-log/2017/10/15/learning-word-embedding.html

Word2Vec - Cbow (Wort zum Kontext)

- Gegeben Kontext “Die”, “Ente”, ”und”, “tanzt””, wollen wir “**quakt**” vorhersagen.
  - “*Die Ente* **quakt** *und tanzt.*” (Fenstergröße 5)
  - Kontext: [“Die”, “Ente”, ”und”, “tanzt”]; Target: [“**quakt**”]
  - Trainingsdaten: X= [“Die”, “Ente”, ”und”, “tanzt”], Y=[“**quakt**”]



https://lilianweng.github.io/lil-log/2017/10/15/learning-word-embedding.html

Word Embeddings, Abstände, Gensim

- Mit Word Embeddings lassen sich nun Abstände berechnen
  - Bsp.: “Ente” und “Gans” haben einen geringeren Abstand als “Ente” und “Auto”, da sie in *Texten* öfters mit den gleichen Kontext Wörtern vorkommen
- Gensim* ist ein Python package, das das Handling von Wortvektoren vereinfacht.
- Beispiel:

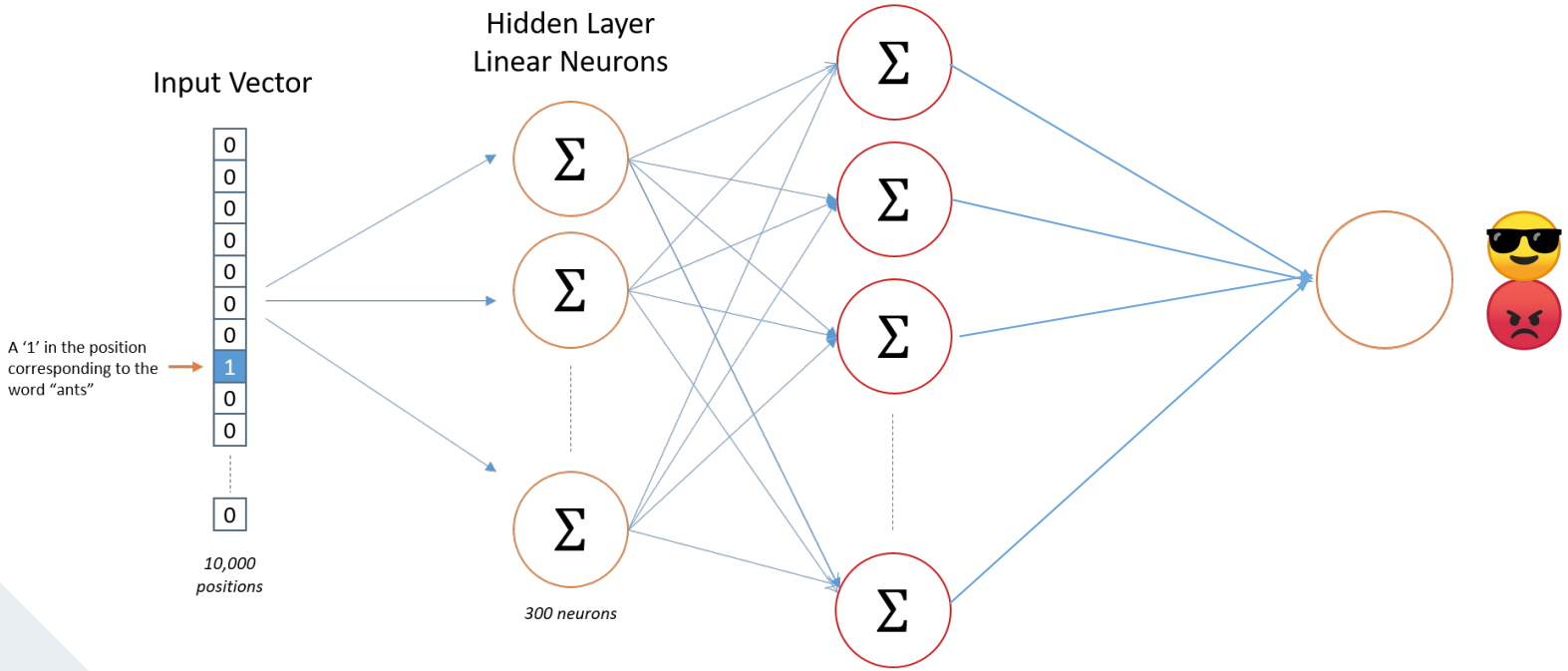
```
from gensim.models import Word2Vec
model = Word2Vec.load("GoogleNews-vectors-negative300.bin'")
model.similarity('germany', 'france')
```


*https://radimrehurek.com/gensim/

Klassifikation
Sentiment Analyse

Sentiment Analysis

Binäre Klassifikation mit BoW



basierend auf http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/

inovex


colab

Google Colab

- In GCP gehosteter Jupyter Notebook Service
- Kollaborative Arbeit (ähnlich GDoc) möglich
- Frei nutzbar (inklusive GPU/TPU!)
  - → Ressourcen sind limitiert
  - → GPU/TPU Runtime sparsam nutzen!
- Notebooks und Daten liegen im Google Drive

Feedback

https://forms.gle/zDjawTGbbs1Z8ryXA

Vielen Dank

inovex GmbH
Ludwig-Erhard-Allee 6
76131 Karlsruhe