Anwendungen der KI

– Sommersemester 2018 –

Kapitel 09: Wissensgraphen und Informationsextraktion

Prof. Adrian Ulges
B.Sc. {Angewandte, Medien-, Wirtschafts-}informatik
Fachbereiche DCSM
Hochschule RheinMain

Semantik

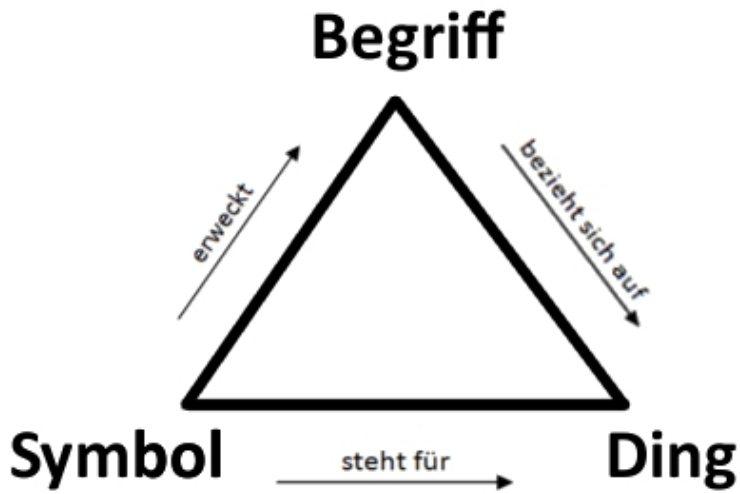
- Wir wollen uns im letzten Abschnitt der Vorlesung mit **semantischen Ansätzen** befassen.
- **Semantik** ist die Lehre von der **Bedeutung** von (Morphemen), Wörtern und Sätzen.

Was verstehen wir unter 'Bedeutung'?

Das semiotische Dreieck

Semantik ist **komplizierter** als gedacht[1]:

- Die **Bedeutung** eines Wortes (bzw. Satzes) ist ein "Konzept, das eine mentale Beschreibung einer bestimmten Art von Entitäten (bzw. Situationen) bereitstellt."

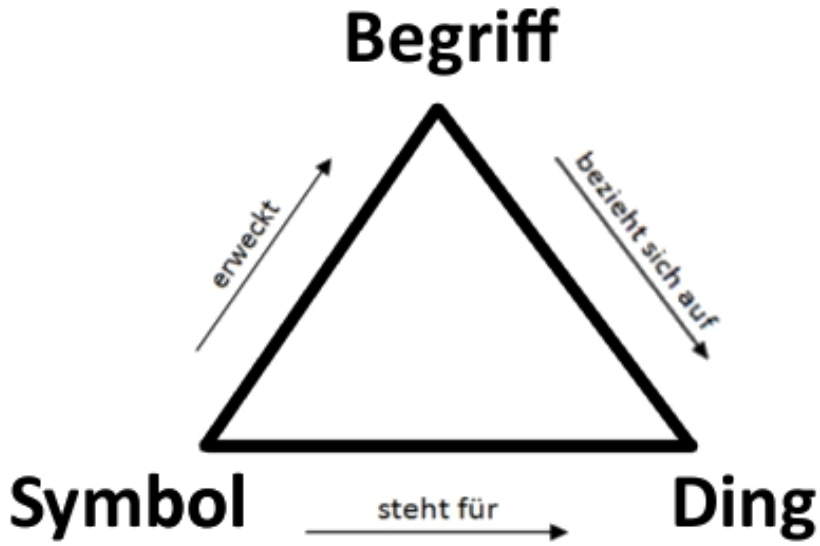


Beispiel

> *"I know that book."*

[1] http://fak1-alt.kgw.tu-berlin.de/call/linguistiktutorien/semantik/index.html

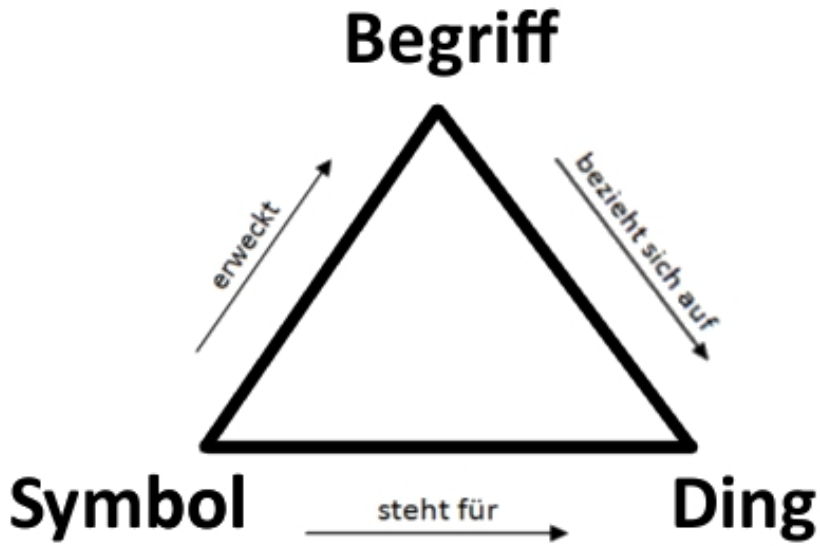
Das semiotische Dreieck



Grundproblem: Die Beziehung zwischen Symbol *(das Wort "Buch")* und Objekt *("Ding")* ist **zweischrittig**.

1. Das **Symbol** *erweckt* die Assoziation mit einer abstrakten Vorstellung / einem **"Konzept" oder "Begriff"**.
2. Das **Konzept** bezieht sich auf ein **Objekt/Ding** der realen Welt.

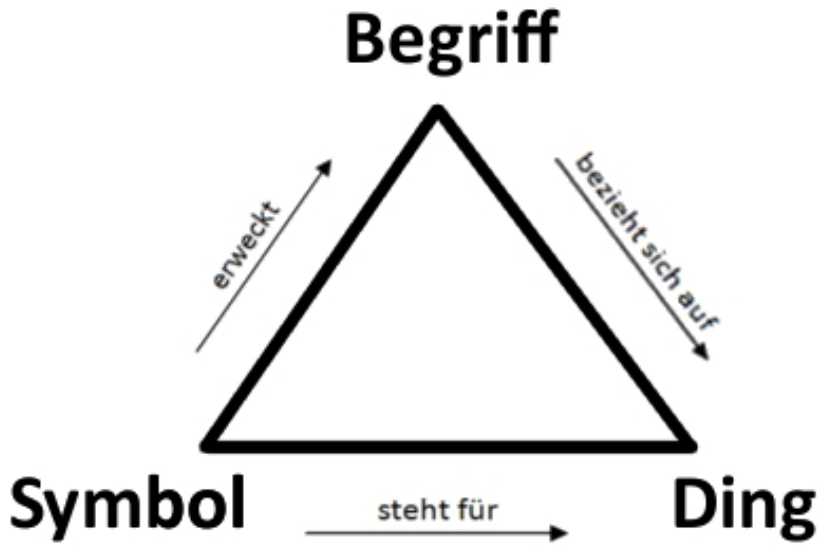
Das semiotische Dreieck



Schritt 1: Symbol → Begriff

- Welche Symbole *(Worte)* stehen wofür? *(u.a. parts-of-speech)*
- Wie sind Zeiten/Fälle? ("know" vs. "will know") *(Grammatik)*
- Welche Worte bilden Gruppen, wie "the book"? Wo sind Subjekt, Prädikat und Objekt? *(Syntax)*
- **Mehrdeutigkeiten** entstehen z.B. durch Homonyme ("bank") und mehrdeutige Syntax.

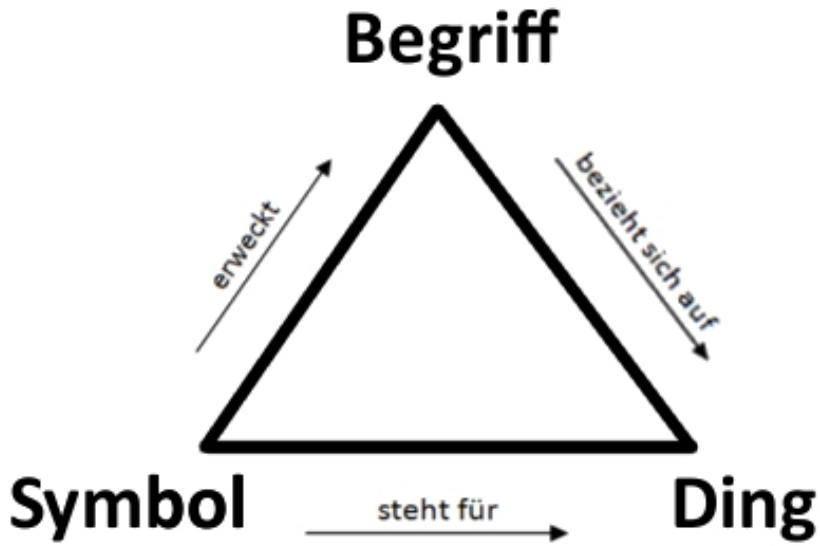
Das semiotische Dreieck



Schritt 2: Begriff → Ding

- Das Symbol ("the book") befindet sich nun als **Vorstellung** in unseren Köpfen *("Ding aus Papier mit Pappdeckel, enthält Roman oder Kochrezepte")*.
- Welches **weltliche Ding** ist mit "the book" gemeint?

Das semiotische Dreieck



Schritt 2: Begriff → Ding

- Dies hängt vom textlichen Kontext ab!

I've never seen 'The Godfather', but I've read the book.

- Hier entstehen **Mehrdeutigkeiten** durch **subjektive Interpretationen**, **Unvollständigkeiten**, die Auflösung **relativer Anschlüsse**, etc.

Semantik

Das **traditionelle Ziel** der Semantik im **Informatik-Kontext** lautet: Überführe Äußerungen in natürlicher Sprache in eine **logische Form**.

Beispiel: Wie repräsentieren wir hier die Bedeutung?

> *"I carried a watermelon."*
>
> (???)

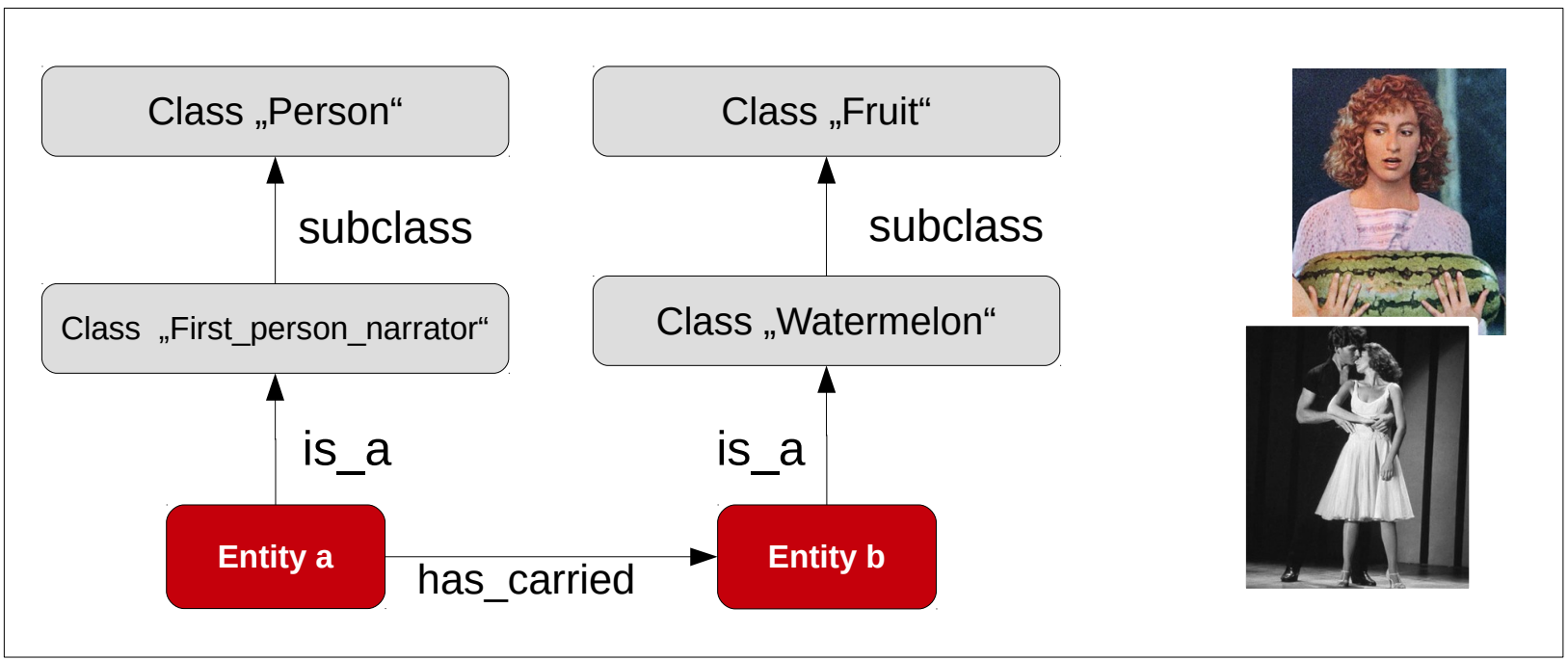
Repräsentation in logischer Form

Semantik

"I carried a watermelon."

(Jennifer Grey / Dirty Dancing)

Repräsentation als Wissensgraph



Outline

Semantik im QA?

Wieso sind Wissensgraphen für unser Projekt (QA) relevant?

Weil wir Fakten in Wissensgraphen "nachschlagen" können
(z.B. indem wir nach Mustern suchen)

Question:
The granddaughter of which actor starred in the movie E.T.?

Pattern *(in SPARQL-Syntax)*:

```
(?x <acted-in> <E.T.>)
(?y <is-a> <actor>)
(?x <granddaughter-of> ?y)
```

Problem

- Wie bilden wir Terme in natürlicher Sprache *(Symbole)* auf **Knoten** und **Kanten** unseres Wissensgraphen ab?
- **Im Beispiel**: Woher erkennen wir, dass sich die Frage "starred in" auf die Schauspieler-Film-Relation `<acted-in>` bezieht?

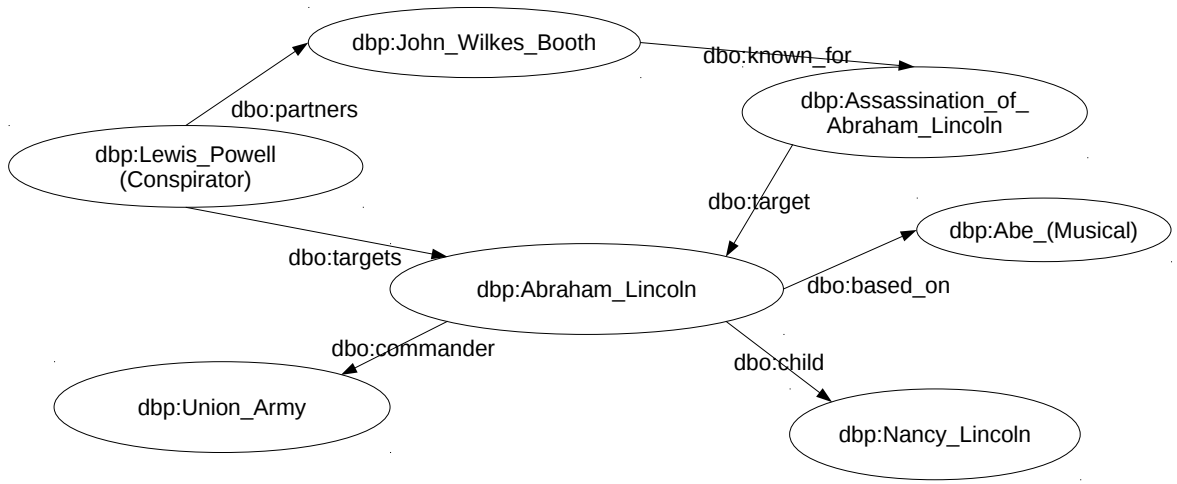
Semantik im QA? (cont'd)

Einfacherer Ansatz: **Unscharfe Suche** nach Kandidaten

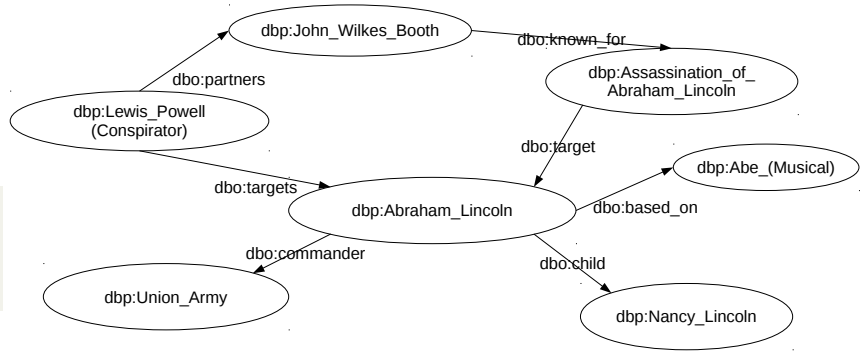
1. Finde **Entitäten** aus der Frage in der Wissensbasis.
2. Durchsuche das **lokale Umfeld** der Entitäten im Wissensgraph nach potenziellen Antworten.
3. Filtere diese nach **Relationen**/**Begriffen** aus der Frage.

Beispiel (Wissensgraph DBPedia)

> *Who murdered Abraham_Lincoln?*



Semantik im QA? (cont'd)



Who murdered Abraham Lincoln?

- **Was brauchen wir hierfür?** Wir müssen Entitäten/Knoten im Wissensgraph mit ihren Erwähnungen *(engl. mentions)* in der (natürlichsprachlichem) Frage verknüpfen.
  - (leicht?) `dbp:abraham_lincoln = 'Abraham Lincoln'`
  - (schwer?) `dbp:abraham_lincoln ≈ 'the 16th President'`
- Dasselbe gilt für Relationen (`murdered` ≈ `assassination_of_abraham_lincoln`)

Welche Informationsquellen gibt es hierzu?

- Thesauri *(gleich)* → Synonyme, Unter-/Oberbegriffe
- Embeddings *(z.B. word2vec)* → Termähnlichkeiten *(z.B. Lincoln → President)*
- Wikipedia!

Wikipedia als Informationsquelle I: Mentions

- Eine **Wikipedia-Seite** steht für eine **Entität** (z.B. `https://en.wikipedia.org/wiki/Donald_Trump` = `dbp:donald_trump`)
- Die Seite/Entität wird an anderen Stellen in Wikipedia mit verschiedenen Ankerphrasen **erwähnt** (engl. *backlinks*).

> *In 2018,* **[[Stormy Daniels]]** *became involved in a legal dispute with* **[[the president|Donald Trump]]** *and his attorney ...*

- Wir **parsen** diese Ankerphrasen aus einem **Wikipedia-Dump**:

```
1  RE_LINKS = re.compile(r'\[\[([^][]*)\]\]')
2
3  def digest_wikipage(markup_text):
4      mentions = {}
5      ...
6      for match in re.finditer(RE_LINKS, markup_text):
7          m = match.group(0)
8          entity,mention = m[2:-2].split('\pipe')
9          ...
10         mentions[entity].append(mention)
11
12     ...
13     return mentions
```

Wikipedia als Informationsquelle II: Redirects

- Wenn wir im Browser `en.wikipedia../Donald_J_Trump` eingeben, landen wir bei `en.wikipedia.../Donald_Trump`.
- Solche **Redirects** werden in der Wikipedia händisch gepflegt.
- Sie sind sehr nützlich, denn sie enthalten **synonyme Bezeichnungen** für Wikipedia-Seiten/Entitäten!
- Redirects sind in Wissensgraphen wie DBPedia gespeichert.

Beispiel-Zugriff (in SPARQL-Syntax):

```
select distinct ?c
where {?c
<http://dbpedia.org/ontology/wikiPageRedirects>
<http://dbpedia.org/resource/Donald_Trump> }
```

http://dbpedia.org/resource/Donald_Trup	http://dbpedia.org/resource/Baron_Trump
http://dbpedia.org/resource/Donald_John_Trump,_Sr.	http://dbpedia.org/resource/Trump_Sr.
http://dbpedia.org/resource/Donald_trump	http://dbpedia.org/resource/DonaldJTrump.com
http://dbpedia.org/resource/Barron_Trump	http://dbpedia.org/resource/Donald_Trump's
http://dbpedia.org/resource/The_Donald	http://dbpedia.org/resource/Danold_Trump

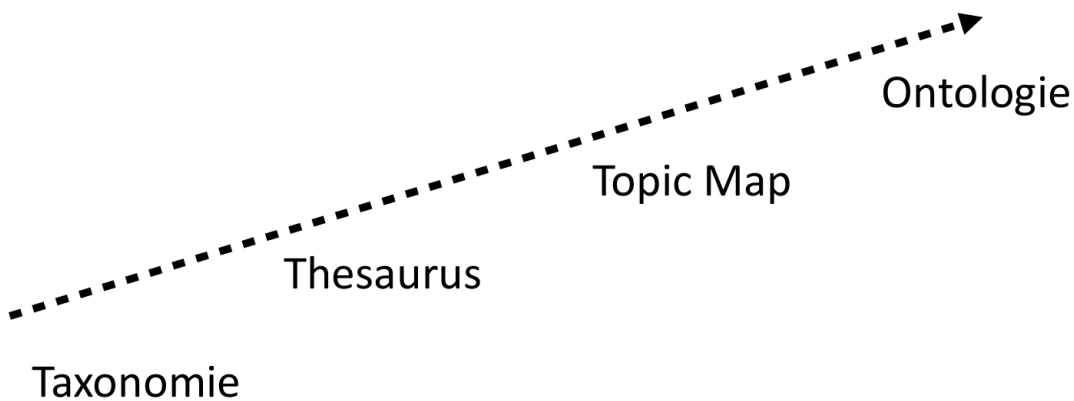
Outline

Semantische Repräsentation: Wissensgraphen

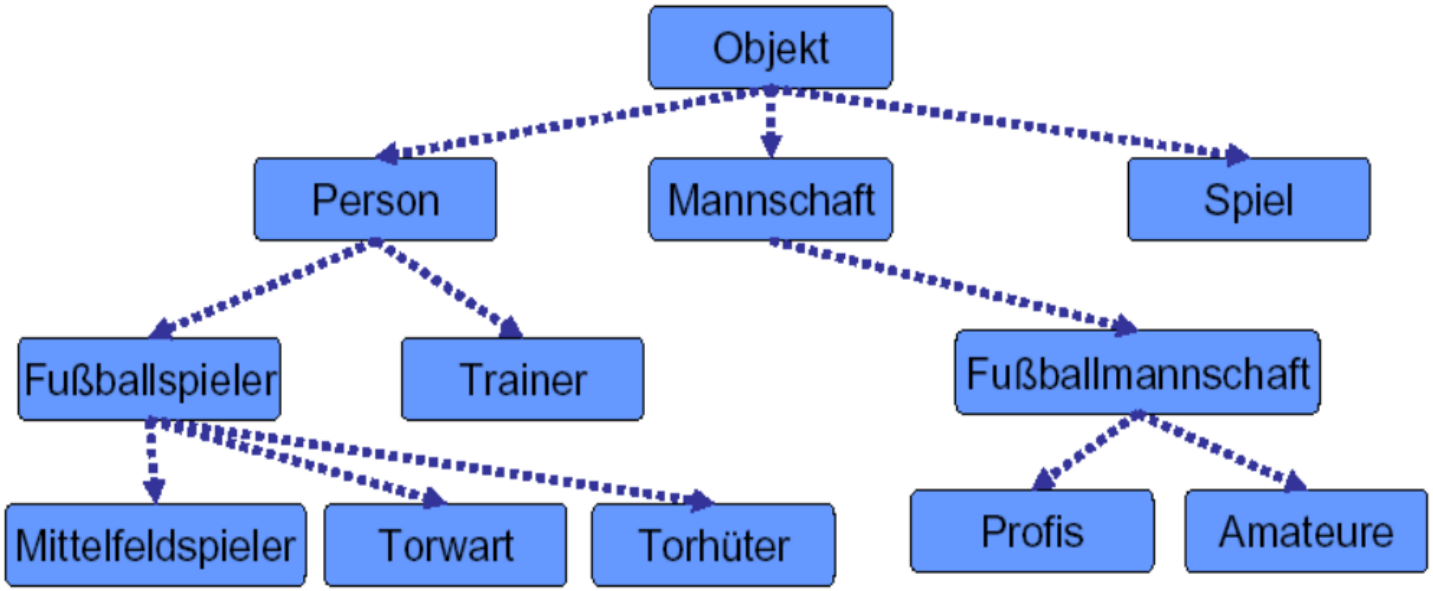
Wir haben bereits gesehen, dass Wissen gerne in Form von **Wissensgraphen** gespeichert wird.

- In diesem Graphen stellen die **Knoten** Konzepte und Dinge dar, und **Kanten** die **Beziehungen** zwischen ihnen.
- Streng genommen müssen wir (siehe unten) zwischen zwei Arten von Knoten unterscheiden:
  1. **Lemma (Symbol)**: Ein Lemma ist eine Wort-Grundform *(typischer Weise: Ein Eintrag in einem Wörterbuch)*.
  2. **Konzept**: Die Wortbedeutung *(engl. word sense)*: "a discrete representation of an aspect of a word's meaning".
     In Wordnet (s.u.) werden Konzepte z.B. **Synsets** genannt.

Wir unterscheiden verschiedene **Ausbaustufen**:

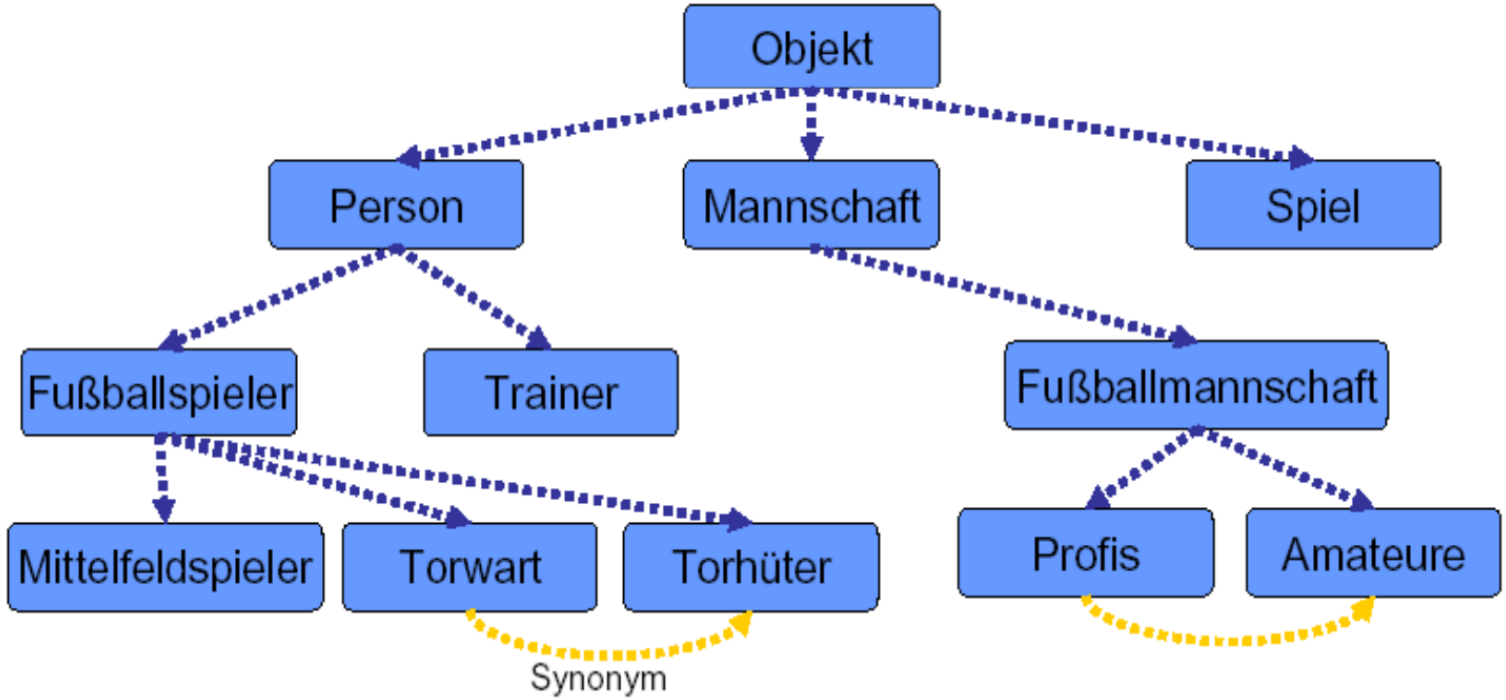


1. Taxonomien



- Taxonomien kennen nur eine Beziehung:
  Die **Hypernym(=Subklassen)-Beziehung**.
- Taxonomien stellen also **Begriffshierarchien** dar.
  Sie erlauben die Segmentierung / Klassifikation der Objektwelt.
- Taxonomien bilden **Gerichtete azyklische Graphen**
  (engl. directed acyclic graphs, DAGs).

2. Thesauri



- Thesauri beinhalten die **Terminologie** einer **Domäne** ("kontrolliertes Vokabular").
- Sie enthalten wenige, **fest vorgegebene** Beziehungen.

Beziehungen im Thesaurus

1. **Synonyme**: Begriffe mit gleichem (oder sehr ähnlichem) Bedeutungsumfang *(couch/sofa, big/large, vomit/throw up)*

2. Thesauri

Beziehungen im Thesaurus (cont'd)

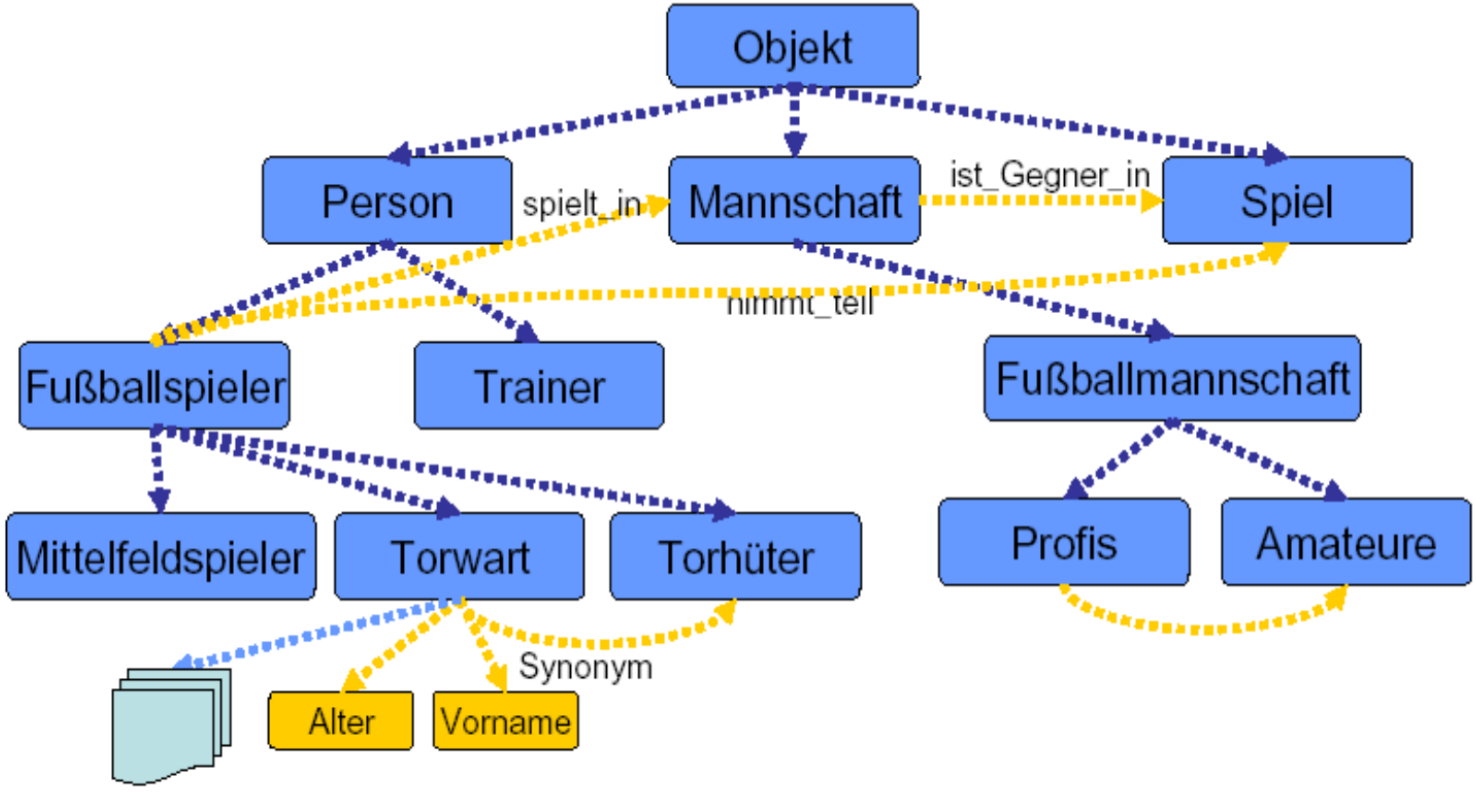
2. **Homonyme**: Begriffe mit gleicher Form aber unterschiedlichen Bedeutungen (Konzepten).
   - $\texttt{bank}_1$: Finanzinstitut, $\texttt{bank}_2$: Sitzgelegenheit
   - Homographen = gleiche Schreibweise *(bat)*,
     Homophone = gleiche Aussprache *(piece/peace)*
3. **Antonyme**: Konzepte, die mit Bezug auf mindestens ein Merkmal das **Gegenteil** sind, z.B. als Enden einer Skala oder als Umkehrungen.
   - `dark`/`light`, `short`/`long`, `up`/`down`, `black`/`white`, ...
4. **Hyponyme**: Konzept A ist ein *Hyponym* von Konzept B, wenn A **spezifischer** ist als B
   - `car` ist ein Hyponym von `vehicle`, `mango` von `fruit`, ...
5. **Hypernyme**: Konzept B ist ein *Hypernym* von Konzept A, wenn B **allgemeiner** ist als A
   - `vehicle` ist ein Hypernym von `car`, ...

2. Thesauri

Beziehungen im Thesaurus (cont'd)

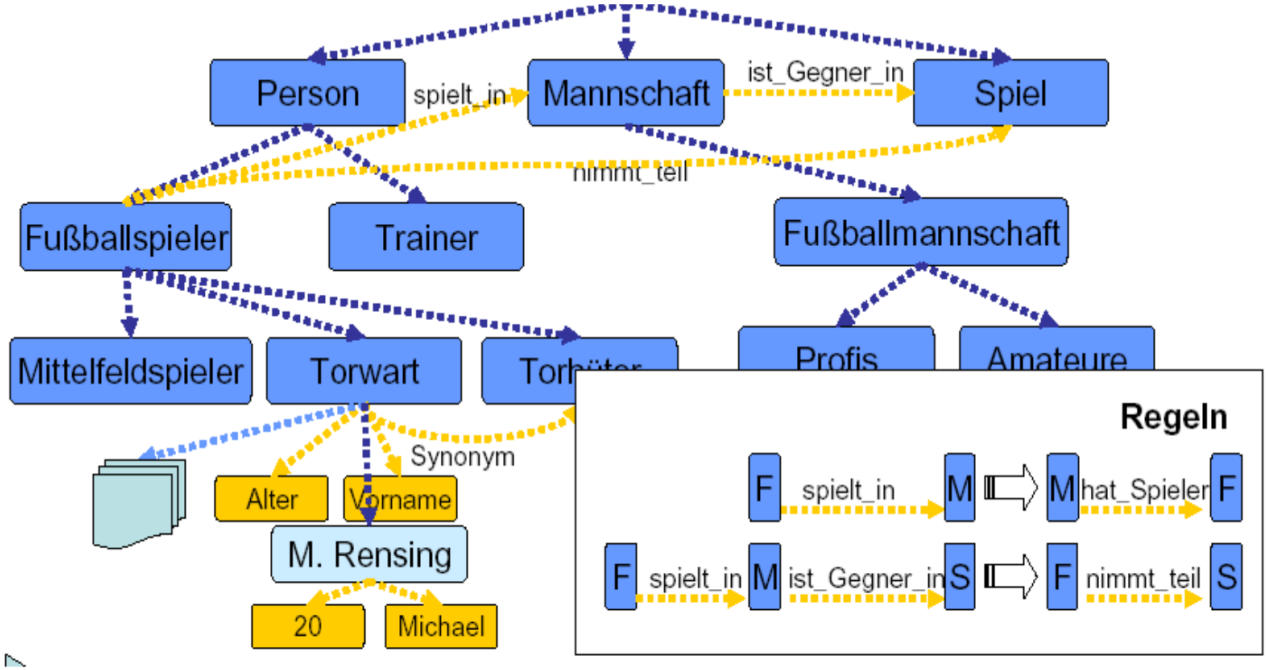
6. **Meronyme**: Konzept A ist ein *Meronym* von B, wenn A als **Teil** (part-of) von B auftreten kann.
   - `finger` ist ein Meronym von `hand`.
7. **Holonyme**: Konzept B ist ein *Holonym* von A, wenn A als **Teil** (part-of) von B auftreten kann.
   - `car` ist ein Holonym von `tire`.

3. Topic Maps



- enthalten Topics (Konzepte) mit **vielen, freien** Relationen
- Fokus auf Visualisierung/Darstellung, **nicht** auf automatisierter Verarbeitung
- keine Inferenz, keine Integritätschecks, keine Vernetzung / Standardisierung.

4. Ontologien



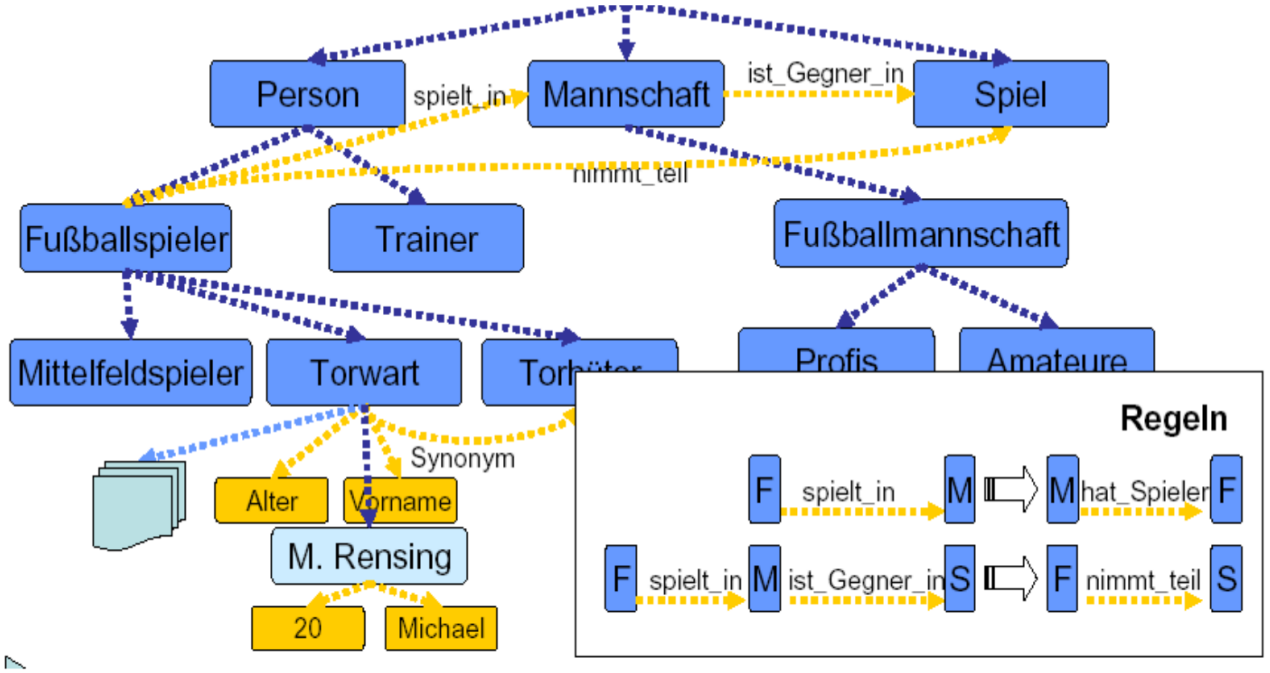
"A specification of a representational vocabulary for a shared domain of discourse – definitions of classes, relations, functions, and other objects – is called an ontology."

(Gruber, T. R. A. Translation Approach to Portable Ontology Specifications / 1993)

"An ontology is a formal, explicit specification of a shared conceptualization"

(Studer, Benjamins, Fensel, 1998)

4. Ontologien



- **"Konzeptualisierung"**: Welche Konzepte existieren in der Welt/Domäne ?
- **"shared"** (kollaborativ) + **"formal"** (maschinen-verarbeitbar)
- Entitäten und Relationen sind **frei spezifizierbar**
- **Inferenz- und Integritätsregeln**
  (z.B. $STUDENTS \cap TEACHERS = \{\}$)

Outline

WordNet[3]

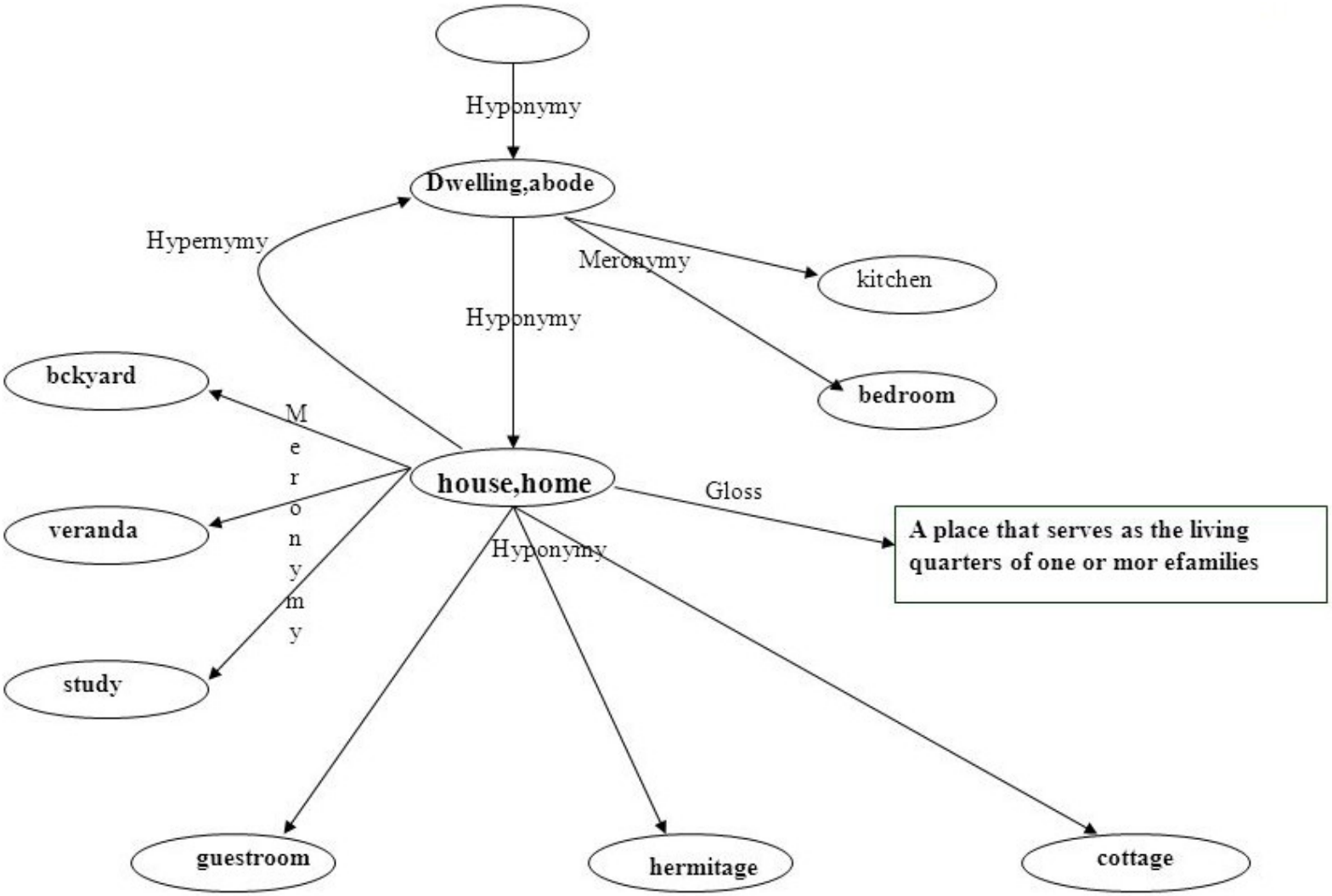
- WordNet ist ein relativ kleiner, aber **sehr bekannter Wissensgraph** für Englisch[2].
- WordNet entspricht der Ausbaustufe **Thesaurus** *(wenige, fest definierte Typen von Relationen, keine Inferenz)*.
- **Wichtiges Feature**: WordNet unterscheidet zwischen **Worten** *(lemmas)* und **Wortbedeutungen** *(senses)*.
- Es gibt ca. 117,000 **Konzepte**, oder **Synsets**. Diese Synsets sind mit ihren **Lemmas** verknüpft.
- WordNet ist in Python (`nltk`) verfügbar!

```
1 from nltk.corpus import wordnet as wn
2
3 syn = wn.synsets("hello")
4 # > [Synset('hello.n.01')]
5
6 syn = wn.synsets("world")
7 # > [Synset('universe.n.01'), Synset('world.n.02'), Synset('
    world.n.03'), Synset('earth.n.01'), Synset('populace.n
    .01'), Synset('global.s.01'), ... ]
```

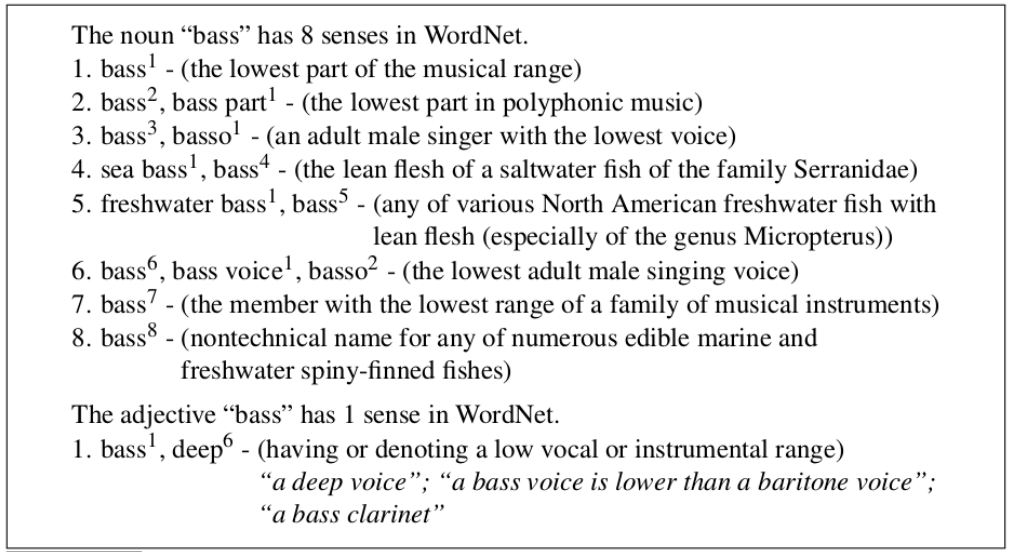
[2] für Deutsch z.B. GermaNet, http://www.sfs.uni-tuebingen.de/GermaNet/

[3] https://wordnet.princeton.edu/

WordNet: Illustration Bild: [1]



WordNet: Synsets vs. Lemmata Bild: [4]



The noun "bass" has 8 senses in WordNet.
1. bass[1] - (the lowest part of the musical range)
2. bass[2], bass part[1] - (the lowest part in polyphonic music)
3. bass[3], basso[1] - (an adult male singer with the lowest voice)
4. sea bass[1], bass[4] - (the lean flesh of a saltwater fish of the family Serranidae)
5. freshwater bass[1], bass[5] - (any of various North American freshwater fish with lean flesh (especially of the genus Micropterus))
6. bass[6], bass voice[1], basso[2] - (the lowest adult male singing voice)
7. bass[7] - (the member with the lowest range of a family of musical instruments)
8. bass[8] - (nontechnical name for any of numerous edible marine and freshwater spiny-finned fishes)

The adjective "bass" has 1 sense in WordNet.
1. bass[1], deep[6] - (having or denoting a low vocal or instrumental range)
"a deep voice"; "a bass voice is lower than a baritone voice"; "a bass clarinet"

Figure 17.1 A portion of the WordNet 3.0 entry for the noun *bass*.

```
1  from nltk.corpus import wordnet as wn
2
3  synsets = wn.synsets("bass")
4
5  for synset in synsets:
6      print synset.definition()
7
8  # > the lowest part of the musical range
9  # > the lowest part in polyphonic music
10 # > an adult male singer with the lowest voice
```

WordNet: Relationen Bild: [4]

Relation	Also Called	Definition	Example
Hypernym	Superordinate	From concepts to superordinates	*breakfast*^{1} → *meal*^{1}
Hyponym	Subordinate	From concepts to subtypes	*meal*^{1} → *lunch*^{1}
Instance Hypernym	Instance	From instances to their concepts	*Austen*^{1} → *author*^{1}
Instance Hyponym	Has-Instance	From concepts to concept instances	*composer*^{1} → *Bach*^{1}
Member Meronym	Has-Member	From groups to their members	*faculty*^{2} → *professor*^{1}
Member Holonym	Member-Of	From members to their groups	*copilot*^{1} → *crew*^{1}
Part Meronym	Has-Part	From wholes to parts	*table*^{2} → *leg*^{3}
Part Holonym	Part-Of	From parts to wholes	*course*^{7} → *meal*^{1}
Substance Meronym		From substances to their subparts	*water*^{1} → *oxygen*^{1}
Substance Holonym		From parts of substances to wholes	*gin*^{1} → *martini*^{1}
Antonym		Semantic opposition between lemmas	*leader*^{1} ⟺ *follower*^{1}
Derivationally Related Form		Lemmas w/same morphological root	*destruction*^{1} ⟺ *destroy*^{1}

Figure 17.2 Noun relations in WordNet.

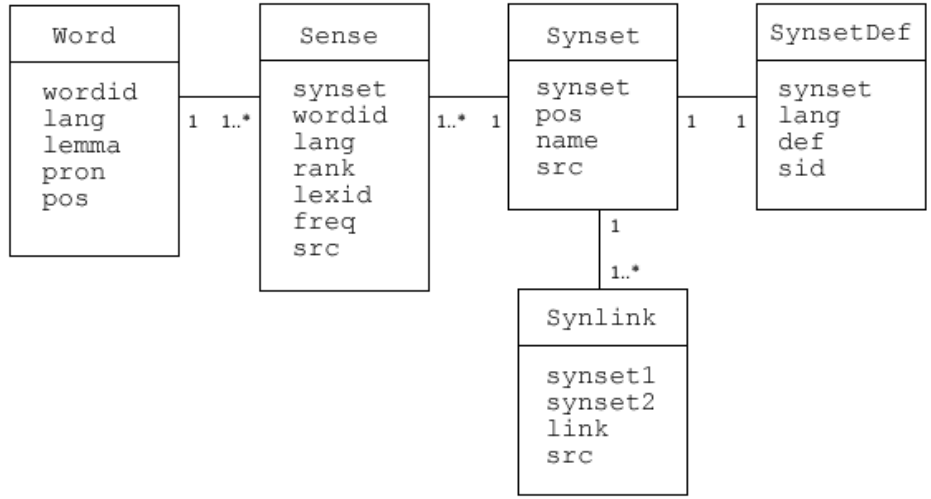
Relation	Definition	Example
Hypernym	From events to superordinate events	*fly*^{9} → *travel*^{5}
Troponym	From events to subordinate event (often via specific manner)	*walk*^{1} → *stroll*^{1}
Entails	From verbs (events) to the verbs (events) they entail	*snore*^{1} → *sleep*^{1}
Antonym	Semantic opposition between lemmas	*increase*^{1} ⟺ *decrease*^{1}
Derivationally Related Form	Lemmas with same morphological root	*destroy*^{1} ⟺ *destruction*^{1}

Figure 17.3 Verb relations in WordNet.

```
1  from nltk.corpus import wordnet
2
3  # president of a republic
4  s = wn.synset("president.n.03")
5
6  for i in range(10):
7      s = s.hypernyms()[0]
8      print (s.name(),
9             s.definition())
10
11 # head_of_state.n.01    : the chief public representative of
       a country ...
12 # representative.n.01   : a person who represents others
13 # negotiator.n.01       : someone who negotiates ...
14 # communicator.n.01     : a person who communicates with
       others
15 # person.n.01           : a human being
16 # causal_agent.n.01     : any entity that produces an effect
       ...
17 # physical_entity.n.01 : an entity that has physical
       existence
18 # entity.n.01           : ...
```

WordNet: Schema

Bild: [2]




```
1  from nltk.corpus \
2      import wordnet as wn
3  from nltk.corpus \
4      import wordnet_ic
5
6  dog = wn.synset('dog.n.01')
7  cat = wn.synset('cat.n.01')
8  mouse = wn.synset('cat.n.01')
9  car = wn.synset('car.n.01')
10
11 # P(c) from brown corpus
12 brown_ic = wordnet_ic.ic('ic-brown.dat')
13
14 print 'dog <-> cat   :', wn.path_similarity(dog, cat), '/',
       dog.res_similarity(cat, brown_ic)
15 print 'dog <-> mouse :', wn.path_similarity(dog, mouse), '/'
       , dog.res_similarity(mouse, brown_ic)
16 print 'dog <-> car :  ', wn.path_similarity(dog, car), '/',
       dog.res_similarity(car, brown_ic)
17
18 # dog <-> cat   : 0.20 / 7.91
19 # dog <-> mouse : 0.20 / 7.91
20 # dog <-> car   : 0.07 / 1.53
```

Term-Ähnlichkeiten: Pfadähnlichkeit

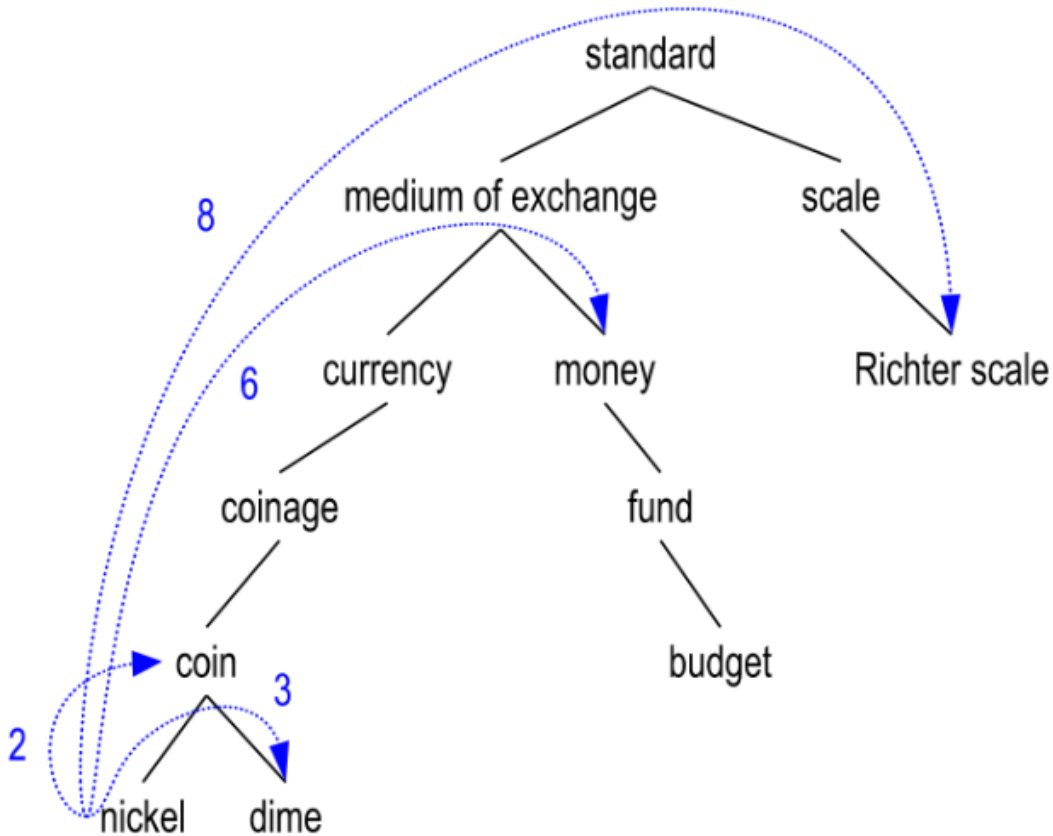
- Wir kennen bereits **Term-Ähnlichkeiten**: syntaktische *(Levenshtein-Distanz)* und statistische *(Word2Vec, PPMI)*
- Thesauri enthalten auch Ähnlichkeiten, z.B. die **Synonym-Relationen**. Diese ist aber **binär** (gleich $\leftrightarrow$ ungleich).
- Mit Hilfe von Taxonomien/Thesauri können wir aber auch **unscharfe** Term-Ähnlichkeiten berechnen.
- Zwei Terme sind **ähnlich**, wenn sie mehrere **Bedeutungsfeatures** teilen.
- **Ziel**: Benutze die Verbindungen des Thesaurus als (unpräzises) **Proxy-Maß** für Ähnlichkeit:

car vs. bycicle	nicht direkt verbunden, ähnlich
car vs. tire	direkt verbunden, weniger ähnlich

Term-Ähnlichkeiten im Thesaurus Bild: [3]

Thesaurus-basierte Term-Ähnlichkeiten

- Sind Wörter “dicht beieinander” in der Hypernym-Hierarchie?
- Haben Wörter ähnliche “Glosses” / Definitionen ?



Term-Ähnlichkeiten: Pfadähnlichkeit

- Wir definieren zuerst die Ähnlichkeit von **Konzepten** c_1, c_2.
- Idee: Zwei Konzepte sind ähnlich, wenn Sie in der Hypernym-Hierarchie **dicht beieinander** stehen.
- Wir definieren $pathlen(c_1, c_2)$ als die Anzahl der Kanten im kürzesten Pfad zwischen Knoten c_1 und c_2.
- Dann lautet die **Pfad-Ähnlichkeit** *(engl. path similarity)*

$$sim(c_1, c_2) := \frac{1}{pathlen(c_1, c_2) + 1}$$

- Ein Konzept hat zu sich selbst die maximale Ähnlichkeit 1.
- Für zwei Terme t_1, t_2 messen wir die Ähnlichkeit als die Ähnlichkeit zwischen ihren am besten **ähnlichsten Konzepten**

$$sim^{terms}(t_1, t_2) := max_{c_1 \in senses(t_1), c_2 \in senses(t_2)} \quad sim(c_1, c_2).$$

Pfadähnlichkeit: Verbesserungen

- **Nachteil** der Standard-Pfadähnlichkeit: Alle Kanten sind **gleichgewichtet**.
- **Beispiel**: Der Schritt von `medium_of_exchange` zu `standard` ist eigentlich viel größer als der Schritt von `nickle` zu `coin`.
- **Beobachtung**: Schritte weiter oben in der Hierarchie sollten teurer sein *(starke Abstraktion)*.

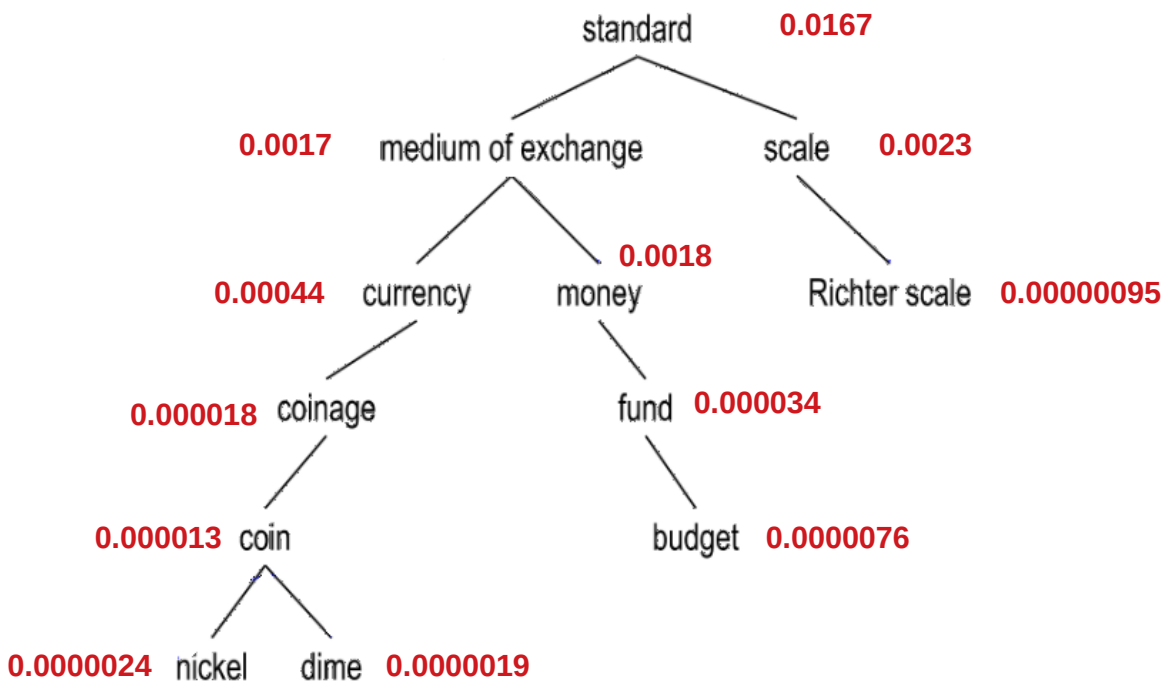
Resnik-Ähnlichkeit

- **Idee (Informationstheorie)**: **Lerne** anhand eines **Textkorpus** die Kosten für teure Schritte.
- Es sei $P(c)$ die **Wahrscheinlichkeit**, dass ein zufällig ausgewähltes Wort (Nomen) des Korpus eine Instanz von Konzept c ist.

Resnik-Ähnlichkeit

Beobachtungen

- **Alle Wörter** stehen für das **Wurzelkonzept**: $P(\texttt{entity}) = 1$
- Je **weiter unten** ein Konzeptknoten in der Hierarchie, desto **geringer** seine Wahrscheinlichkeit: $P(c) \leq P(father(c)) \forall c$.



Resnik-Ähnlichkeit

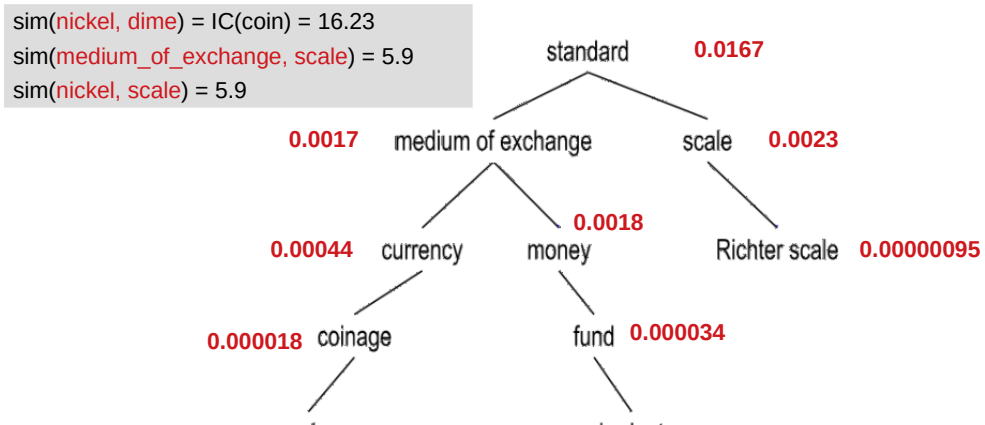
- Wir definieren den **Informationsgehalt** *(engl. information content)* eines Knotens als

$$IC(c) = -\log_2 P(c)$$

- Wir definieren den **LCS** *(lowest common subsumer)* von c_1 und c_2 als und definieren:

$$sim^{resnik}(c_1, c_2) = -log_2 P(LCS(c_1, c_2))$$

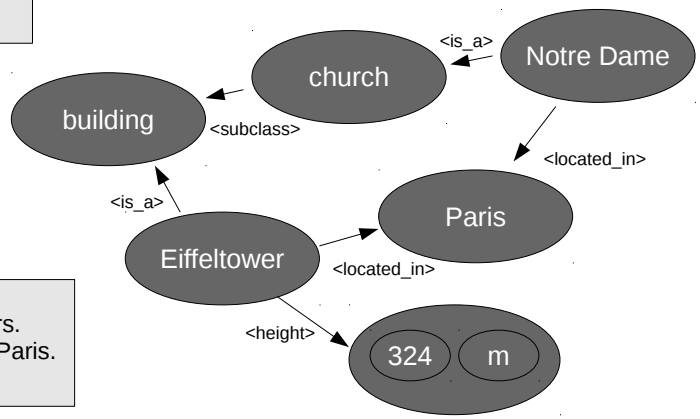
- Dieses Ähnlichkeitsmaß misst: Wie **"speziell"** ist das **gemeinsame Oberkonzept** von c_1 und c_2?

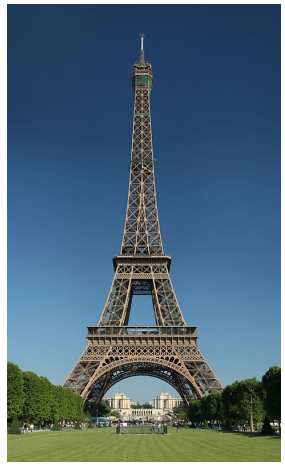


Outline

Sprache → Bedeutung (automatisch) ?

Is the Eiffeltower higher than Notre Dame ?





The Eiffeltower is an iron lattice tower on the Champ de Mars.
With its **height of 324 meters**, **it** is the **tallest** **structure** in Paris.

Die Überführung natürlicher Sprache in eine formale Form ...

- ... wird als **Informationsextraktion** bezeichnet
- ... ist bis heute ein **nicht abschließend gelöstes** Problem.

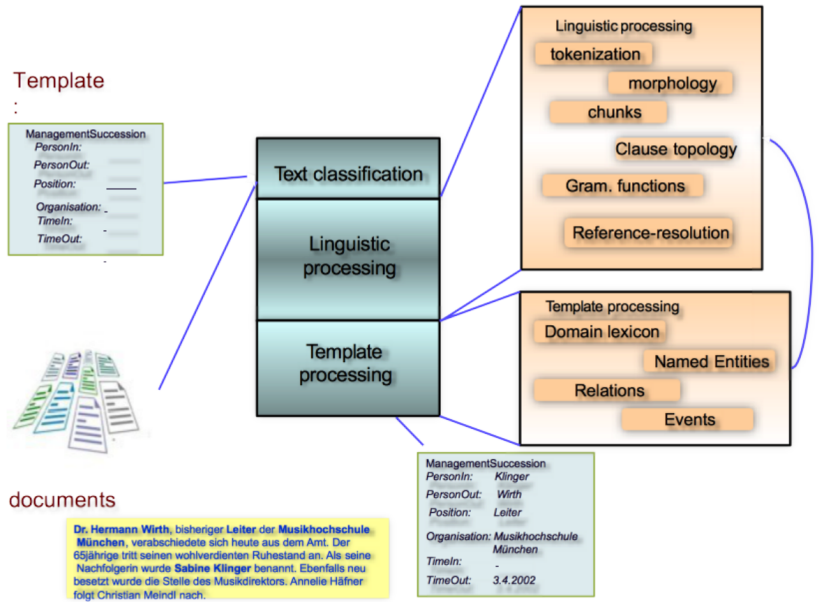
Beispiel: Das Projekt **Read The Web**[4]

cracking_new_year is a monarch	1045	30-mar-2017	96.3
start_search_field is a research project	1046	02-apr-2017	98.8
adrianne_curry is a fashion model	1045	30-mar-2017	99.8

tuna is a type of large_predatory_fish	1046	02-apr-2017
tampa_bay_devil_rays is headquartered in the state or province new_york	1050	19-apr-2017
man is an animal that can develop disability	1046	02-apr-2017

[4] http://rtw.ml.cmu.edu/rtw/

Informationsextraktion



Typischer Aufbau

- Befüllen eines **Musters** (engl. *logical form*)
- Hierzu zahlreiche NLP-Techniken
  (Stemming, Coreference Resolution, ...)
- Erkennung von **Named Entities**
- Erkennung von **Relationen** zwischen Named Entities

Komplexe vs einfache Informationen

Beispiel: Firmenbericht

> *International Business Machines Corporation (IBM or 'the Company') was incorporated in the State of New York on June 16 1911, as the Computing-Tabulating-Recording Co. (C-T-R)...*

Komplexes Template

Company Founding	company	IBM
	location	New York
	date	June 16 1911
	orig_name	Computing-Tabulating-...

In der Regel: Fokus auf Tripel (einfacher)

```
(dbp:IBM, dbp:founding_year, dbp:1911)
(dbp:IBM, dbp:founding_location, dbp:New_Nork)
...
```

Named Entities

- bezeichnen konkrete Dinge *(z.B.* `Donald Trump`*)*
- ... im Gegensatz zu Typen (Auto) und Massenobjekten (Salz)
- **Beispiele**: Personen, Organisationen, Orte, Datums- und Zeitangaben, Messgrößen, Religionen, Buchtitel, Kinofilme und Serien, Medikamentennamen, ...

Named Entity Recognition

- versucht **unbekannte** Entities in Texten zu finden
- haben wir bereits früher kurz behandelt (Kapitel 05)

Named Entity Identification

- versucht **bekannte** Entities in Texten zu finden
- **Meist**: Unscharfes **Matching** gegen Listen bekannter NEs *(engl.* **gazetteers***) (vgl. Informationsquelle Wikipedia', s.o.)*
- Potenziell **Mehrdeutigkeiten**: `Washington` *(Person/Ort?)*, `May` *(Person/Monat/Verb?)*, `1945` *(Jahr/Zeitspanne?)*

Ansätze 1: Gazetteers und Regeln

Gazetteers

- Telefonbücher
- Geonet Names Server[5]
- Wikipedia
- Linked Open Data (LOD)
- ...

Regeln und Templates

```
Rule: Company1                                    from gate.ac.uk
   ( ( {Token.orthography == upperInitial} )+
      {Lookup.kind == companyDesignator}
   ):match
 -->
   :match.NamedEntity = { kind=company, rule="Company1" }
```

Ansätze: Vergleich

Statische Ansätze	**Machine Learning**
benutzen Regeln & Gazetteers	benutzen Sequence Labeling mit BIO-Tags *(vgl. Kapitel 05)*
in der Industrie immer noch weit verbreitet	verbreitet in der Forschung und bei Web-Konzernen
gut geeignet für spezielle Aufgaben	viel besser auf neue Daten / Domänen anpassbar
funktioniert gut mit wenig Daten	funktioniert besser bei komplexen Mustern / Fragestellungen

Named Entity Recognition in Python

Fertige Modelle für NER sind in Python (`nltk`) verfügbar[6]

```
from nltk.tag import StanfordNERTagger

>>> st = StanfordNERTagger('english.all.3class.distsim.crf.ser.gz')
>>> st.tag('Rami Eid is studying at Stony Brook University in NY'.split())

#[('Rami', 'PERSON'), ('Eid', 'PERSON'), ('is', 'O'), ('studying', 'O'),
# ('at', 'O'), ('Stony', 'ORGANIZATION'), ('Brook', 'ORGANIZATION'),
# ('University', 'ORGANIZATION'), ('in', 'O'), ('NY', 'LOCATION')]
```

[6] https://nlp.stanford.edu/software/CRF-NER.shtml

Relations-Klassifikation

In der Regel lernende Ansätze

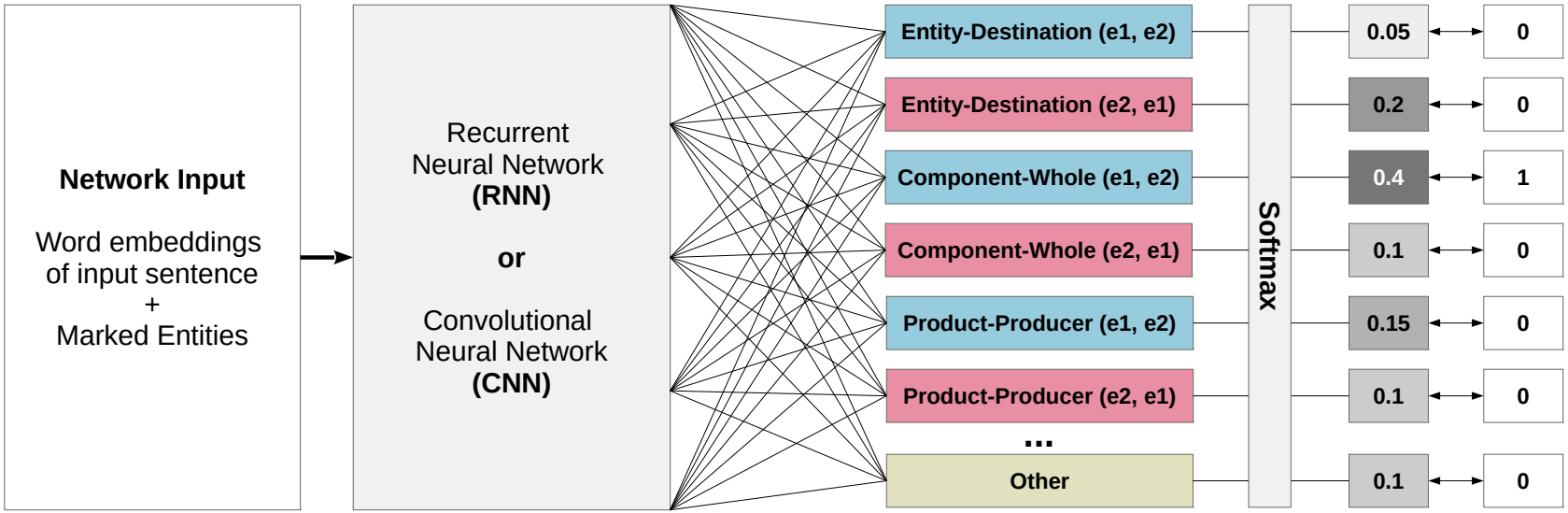
- **Datengrundlage**: Viele **Trainingssätze** mit der Zielrelation, z.B. `dbp:founding_year`

> **IBM began in 1911** *as C-R-T ...*
> *During its* **first four years** *(***1976***-1980),* **Apple** *focused on ...*
> **Google** *was* **founded in 1998** *by Larry Page and ...*
> *The* **IBM company** *was* **launched in 1998** *...*
> *...*

Woher kommen diese Daten?

- **Option 1**: Händisch annotierte Trainingssätze.
- **Option 2**: **Distant Supervision** [5]. Wir crawlen per Suchmaschine Trainingssätze aus dem Web, in denen (`IBM`,`1911`), (`Apple`,`1976`) oder (`Google`, `1998`) vorkommen. Viele *(aber nicht alle)* dieser Sätze enthalten die korrekte Relation. Wir bezeichnen solche Daten als **schwach gelabelt**.

Relationsextraktion mit Neuronalen Netzen



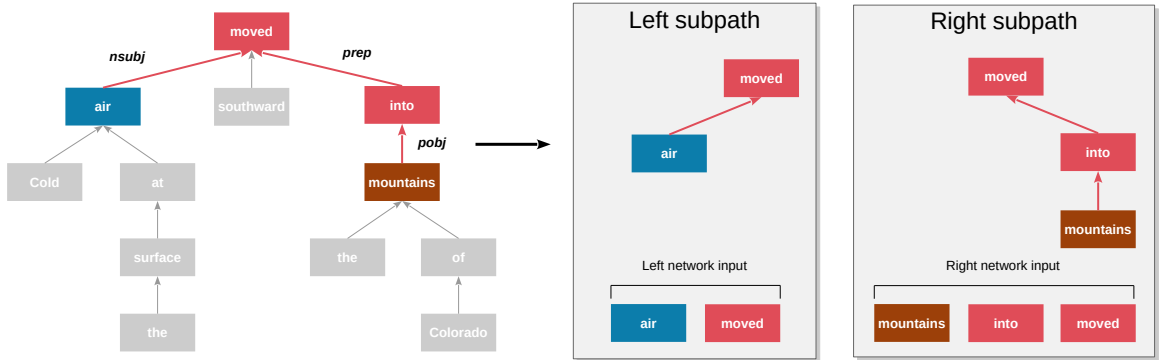
Master-Projekt Markus Eberts und Felix Hamann

- Ein Satz wird ist eine **Sequenz** von Term-Embeddings.
- Zusätzlich sind darin die **Entitäten**, die in Relation zueinander stehen, **markiert**.
- Diese Informationen werden in ein **rekurrentes Netz** (RNN) oder in ein **convolutional network** (CNN) gefüttert.
- Die Netze ordnen den Satz einer Relation zu.

Beispiel-Architektur

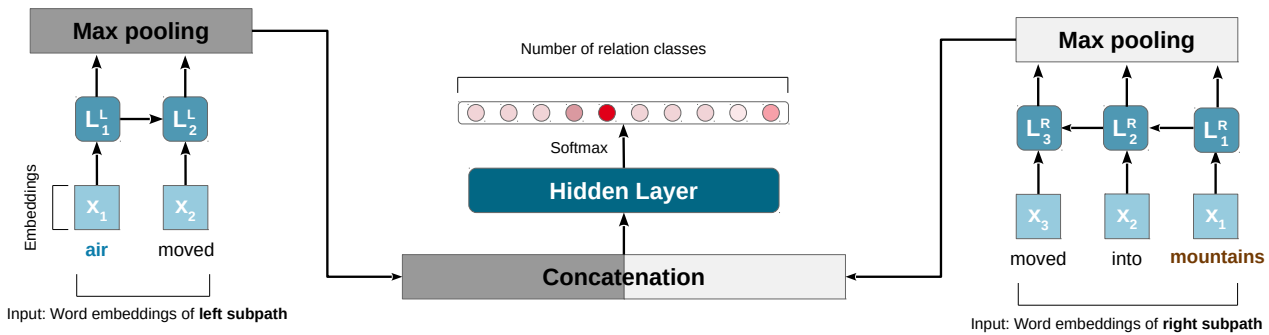
Schritt 1: Vorverarbeitung mit Syntaxbaum

- Bestimme den **Dependenzbaum** des Satzes *(vgl. Kapitel 4)*
- Ermittle den **Shortest Dependency Path** zwischen den Entitäten



Schritt 2: Rekurrentes Netz

- Füttere beide Teilpfade in ein Netz und kombiniere sie.



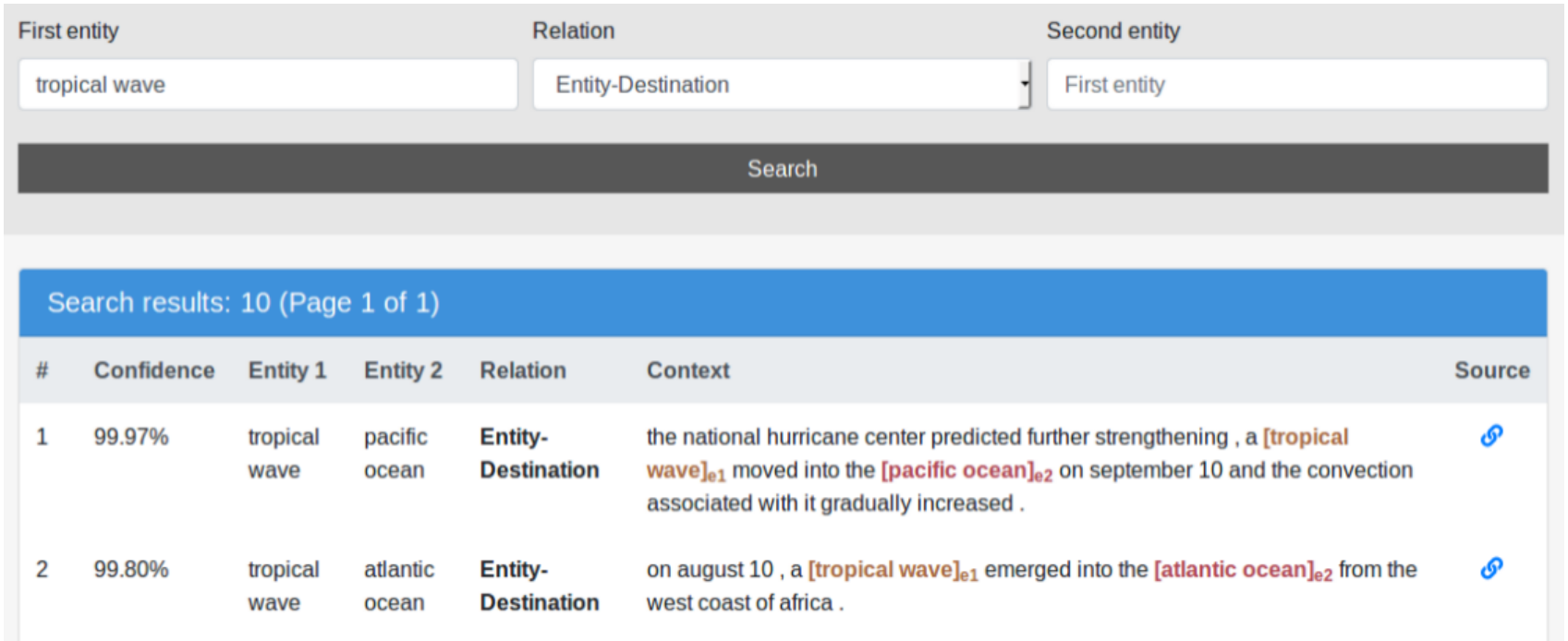
Relationsextraktion mit NNs: Ergebnisse

Beste erreichte Klassifikationsgenauigkeit: $\approx 84\%$
(1200 Sätze, 19 Relations, random guessing entspräche 6%).

0.961598	product producer (straight)	the labyrinth is a large aerial puzzle constructed by the architect daedalus to imprison pandora in the caverns of olympus , home of skorpius and its offspring .
0.953198	product producer (straight)	the music was composed by veteran composer koji kondo , who created new interpretations of the familiar melodies from earlier games as well as entirely new material .
0.947602	product producer (straight)	bastion is an action role-playing video game developed by independent developer supergiant games and published by warner bros. interactive entertainment .
0.984188	member collection (reverse)	between 1970 and 1991 the population of pygmy hippos born in captivity more than doubled .
0.975697	member collection (reverse)	saotherium and choeropsis are significantly more basal than hippopotamus and hexaprotodon , and thus more closely resemble the ancestral species of hippos .
0.973551	product producer (straight)	tales of a sea cow is a 2012 film by icelandic-french artist etienne de france " documenting " a fictional 2006 re-discovery of a population of steller 's sea cows off the coast of greenland .

Relations-"Suchmaschine"

- Web-Interface für **semi-formale Suche** in Wikipedia-Artikeln
- Wir finden mit den neuronalen Netzen Sätze, die bestimmte Entitäten / Relationen enthalten.
- Die Sätze werden nach absteigender **Konfidenz** des neuronalen Netzes gerankt.



References I

[1] Pushpak Bhattacharyya.
Natural Language Processing/Speech, NLP and the Web (Lecture 04: Wordnet and Wordsense Disambiguation, cntd).
http://slideplayer.com/slide/7474656/ (retrieved: May 2018), 2011.

[2] Hideki Shima.
JawJaw: Java Wrapper for Japanese WordNet.
https://www.cs.cmu.edu/~hideki/software/jawjaw/index-en.html, retrieved: May 2018).

[3] Dan Jurafsky.
From languages to information (lecture: Word meaning and word similarity).
https://web.stanford.edu/class/cs124/lec/semlec.pdf (retrieved: May 2018), 2013.

[4] Daniel Jurafsky and James H. Martin.
Speech and Language Processing: An Introduction to Natural Language Processing (3rd Edition Draft Chapters).
2017.

[5] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky.
Distant Supervision for Relation Extraction Without Labeled Data.
In Proc. ACL 09, pages 1003–1011, 2009.