Introduction to Natural Language Processing (NLP) for Digital Humanities

Session 5

Jonas Widmer, University of Bern
WORCK Training School 2, February 2024

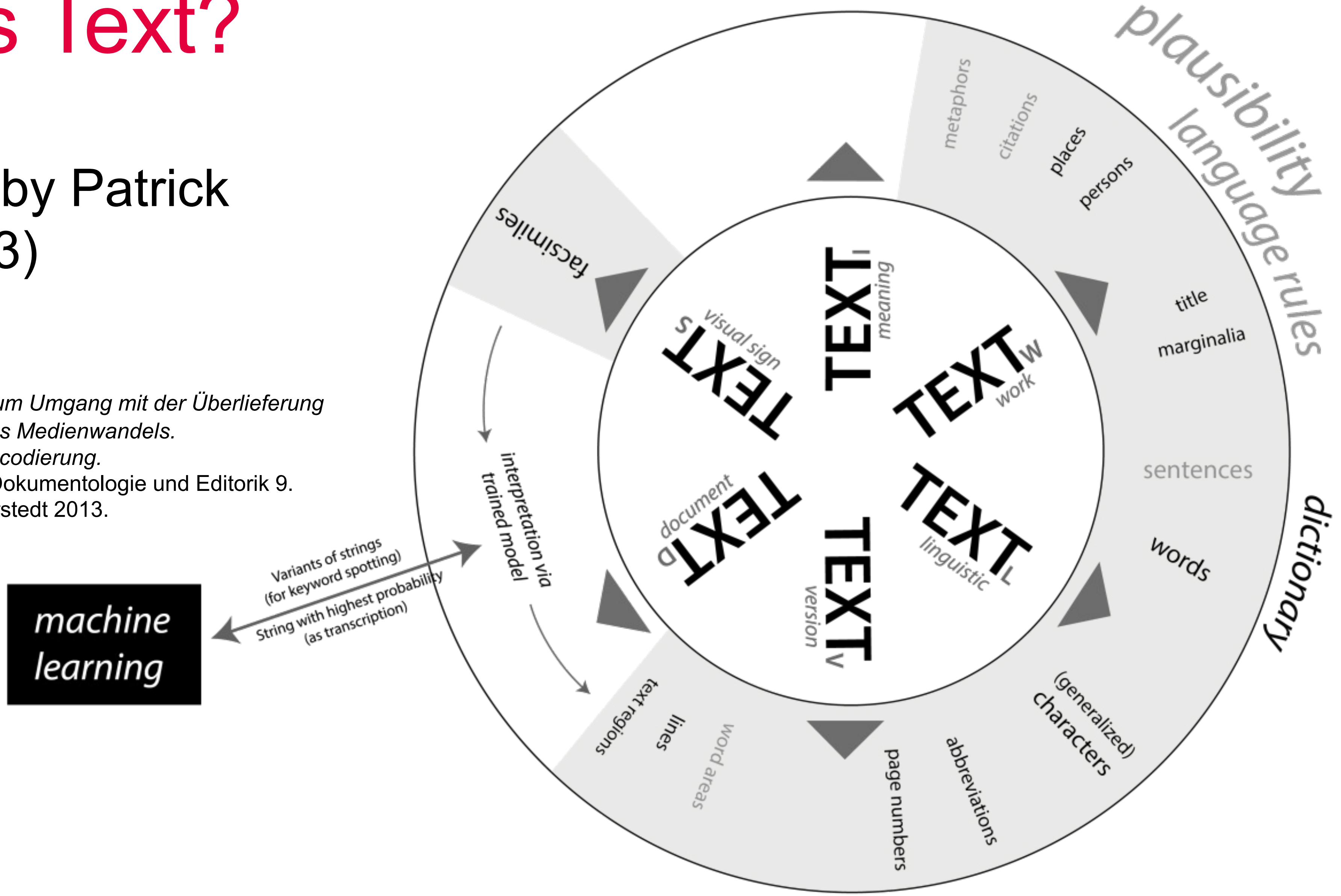
Digital Humanities – Crossing Boundaries

- Developed from Humanities Computing
  → Computer as an auxiliary tool
- (partially) focusing on questions of the humanities with digital means
- Questions, solved with approaches based on computing power
- Like here: Let’s vectorize language! ☺

What is Text?

Text wheel by Patrick Sahle (2013)

Patrick Sahle
Digitale Editionsformen. Zum Umgang mit der Überlieferung unter den Bedingungen des Medienwandels.
Teil 3: Textbegriffe und Recodierung.
Schriften des Instituts für Dokumentologie und Editorik 9.
Books on Demand, Norderstedt 2013.



What is Natural Language Processing?

- Very broad term to describe **methods, which process text**
- **Text is an unstructured data type** – sequential data
- Some subfields are:
  - Information Extraction
  - Document Classification / Comparison
  - Text Generation & Translation

How to translate a sentence?

- *Input*
  The animals were in the pen.
- *Expected Output in German?*
  Die Tiere befanden sich im Pferch.

Let’s use a dictionary

- **Input**
  The animals were in the pen.

- **Output**
  Der/die/das Tiere waren in der/die/das Stift.

Let’s put some rules in place

- For example: Use correct articles/cases.
- **Input**
  The animals were in the pen.
- **Output**
  Die Tiere waren in dem Stift.

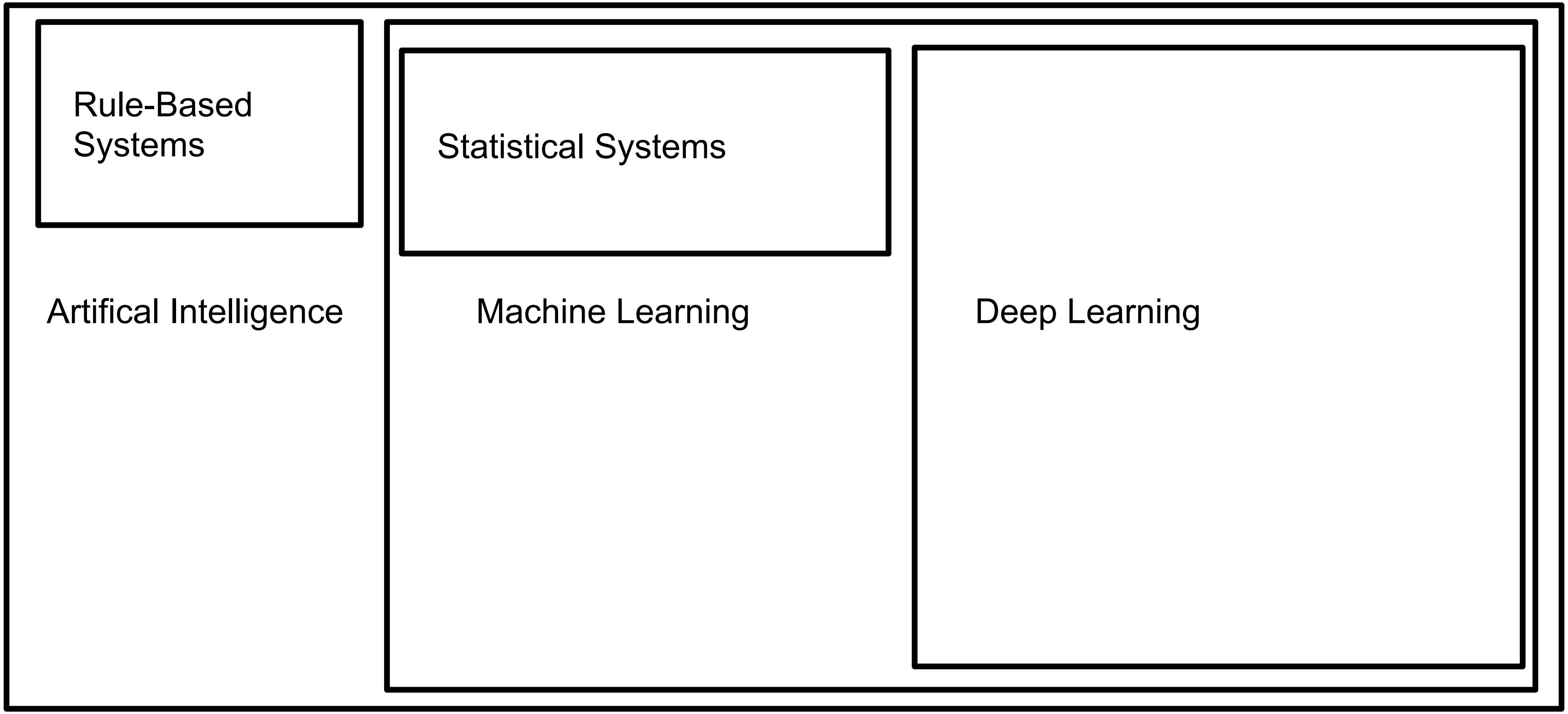
Let’s use probabilities

- “in dem X” → X is more likely a building than a pen for writing.
- “Tiere … X” → In the context of animals, a building is more likely a stable than a pen for writing.

Output
Die Tiere waren in dem Pferch.

But where do we get the probabilities from?

Machine Learning Environment



Quick Glossary

Word Types
Distinct tokens in the corpus

Corpus
Collection of documents

Document
Collection of tokens

Token
- Collection of characters
- Usually base unit

Vocabulary
All word types in the corpus

Rule-based Machine Learning

- Writes machine-behaviour completely by hand
- **Pros:**
  - High level of control
  - No training material needed
- **Cons:**
  - Lots of human work
  - Usually scores low compared to other methods
  - Needs dictionary, lexicons, grammars, and similar.
- Example: Sentiment analysis based on words.

Statistical Machine Learning

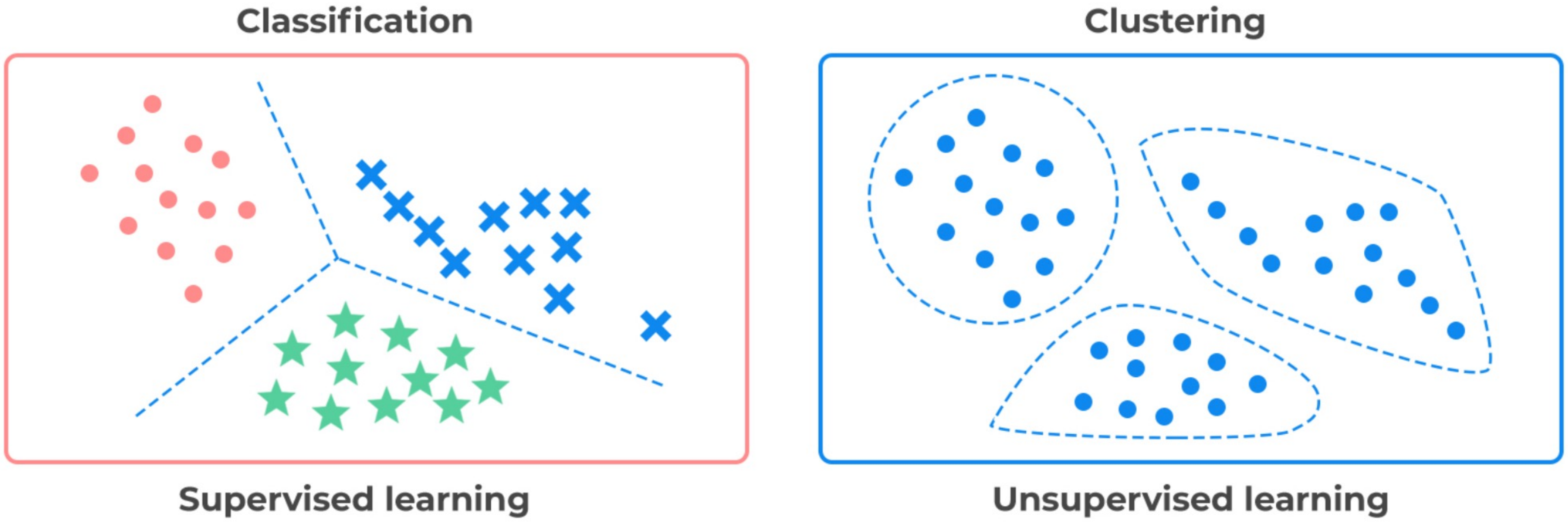
- A model is trained to learn probabilities.
- **Pros:**
  - Still relatively high level of control
  - Small amounts of training data often sufficient
- **Cons:**
  - Usually scores lower than Deep Learning
  - Bad feature engineering can ruin a system
- Example: Predicting the next word based on n advancing words.

Deep Learning

- A neural network is trained to hold the best weights.
- **Pros:**
  - (Usually) Best performance of all methods.
- **Cons:**
  - Large amounts of training data needed
  - Very resource-intensive hardware needed
  - Inner workings often hard to understand
- Example: Large Language Models

Unsupervised vs. Supervised Learning

- **Unsupervised Learning:**
  Detects patterns in data – e.g. grammar of a language
- **Supervised Learning:**
  Assumes a “target” that is learned to generate based on (human) input. E.g. sentiment analysis



Preprocessing

- Preparing the text for further processing by multiple methods

- **Basic use:** Find out where to split a string into tokens.
- **More uses:**
  - Reduce vocabulary size
  - Remove noise
  - Maybe add additional information
  - Split documents into shorter spans to be processed

Sentence-splitting and Tokenization

The representatives of many countries met at the United Nations conference in New York. UN secretary general António Guterres held a speech to open the event; he was focusing on the importance of human rights.

Sentences?

Tokens?

Sentence-splitting and Tokenization

The, representatives, of, many, countries, met, at, the, United, Nations, conference, in, New, York, .

UN, secretary, general, António, Guterres, held, a, speech, to, open, the, event, ;, he, was, focusing, on, the, importance, of, human, rights, .

Sentence-splitting and Tokenization

- **Goal:** Convert the string into lists of sentences and tokens that can be used as input for an ML-pipeline.

- **Split sentences** by either using a ML-model or a rule-based system.
- Basic form of **tokenization** splits at whitespaces and separates punctuation as separate tokens (if not removed).

Normalization

- **Reduce vocabulary size:**
  - More examples per word type
  - Reduces model size & training time

- **Methods:**
  - Casing
  - Lemmatization (and Stemming)
  - Noise Removal
  - Stopword Removal

Casing

- **Truecasing:** “Proper” capitalization, bigger vocabulary
- **Lowercasing:** Make all words lowercase, but losing information for specific tasks (e.g. Named Entity Recognition)

the representatives of many countries met
at the united nations conference in new york.

Lemmatization / Stemming

- With **Lemmatization** we transform words in their base form. E.g. nouns to first person masculine singular, verbs to their infinitives.
- With **Stemming** we remove common prefixes and suffixes (e.g. ”running” to “run”).
- But: We lose information and methods like Byte-Pair Encoding (”splitting” words, e.g. “running” → “run” and “ing”) are preferred.

the representative of many country meet
at the united nation conference in new york .

Noise Removal

- Remove punctuation and special characters (e.g. emojis in social media)
- Aims to get rid of unnecessary noise - but sometimes these characters can carry important information
- Decide on a case-by-case basis if noise removal is necessary
- Also, part of noise removal can be the removal, or replacement, of links, e-mail-addresses, etc.

the representative of many country meet
at the united nation conference in new york

Stop Word Removal

- Remove words that do not carry enough information.
- Modern architectures usually do not require stop word removal as attention algorithms simply ignore what’s not important.
- For some use cases, removal of rare/not relevant words can also be seen as stop word removal (e.g. topic modelling).

~~the~~ **representative** ~~of~~ **many country meet**
~~at~~ ~~the~~ **united nation conference** ~~in~~ **new york**

Enrichment

- Add new features – e.g. labelling, language detection
- Mark collocations / n-grams

~~the~~ **representative** ~~of~~ **many country meet**
~~at~~ ~~the~~ **united_nation conference** ~~in~~ **new_york**

→ **representative many country meet united_nation conference new_york**

Python Libraries for Preprocessing

Natural Language Toolkit (NLTK, nltk.org)

- Good tokenizers and stemmers
- Huge stopword lists (in multiple languages)
- Lots of implementations for other languages
  (e.g. GermaNet from University of Tuebingen)

```
import nltk
from nltk.corpus import stopwords

stops = set(stopwords.words('english'))
print(stops)
```

spaCy (spacy.io)

- Simple usage (got a lot with one function) and performant
- Good models for 18 languages (plus multilingual) – choose between accuracy and efficiency

Vectorization

- **Motivation:**
  - Math with text
  - Calculate similarity of words, documents aso
  - Learn patterns of language
- **But: Must be contextualized and validated (privilege of domain experts).**

Language Models

- Trained models, which can **embed** text (ideally contextualized).
- **Embedding:** Vectorized representation of text
- Per **token, subtoken, character** or **document**, there is a (context-dependent) vector
- **Training with domain-specific corpora**
  - **Unsupervised**
  - **Fine-tuning** or **from scratch**

→ Train a **neural network**, which can vectorize text (of a specific domain).

Language Models

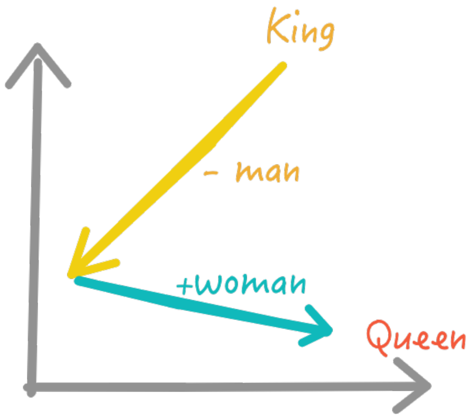
- **Trained vectorization takes care of:**
  - **Similarity by occurrence**
    He had to face the king.
    He had to face the emperor.
  - **Similarity of tokens**
    king
    kingdom

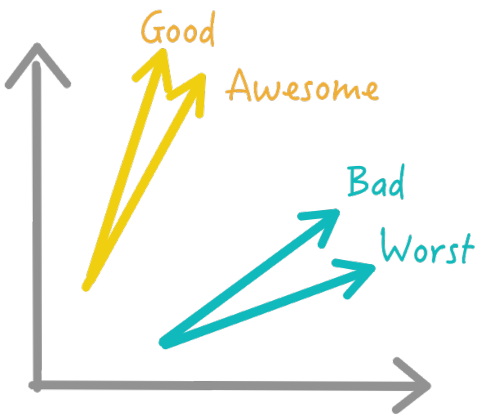
→ Vectors of similar tokens point in a similar direction in space!

Language Models

Word Embedding:
Words are assigned to vectors $v \in \mathbb{R}^n$.

Simplest form of word embedding:
indexing vocabulary (0: Hello, 1: World, …)





Some advantages:

- Arithmetic is possible:
  v('king') - v('man') + v('woman') = v('queen')
- Similarity can be calculated (e.g. Euclidean distance)

Some disadvantages:

- Bias can be learned:
  v('doctor') - v('man') + v('woman') = v('nurse')
- Distortion through polysemy
  e.g. v('bat') → baseball and animal

Problems of not contextualized vectors

Fixed vectors per word/token cannot handle with:

- Shift in language (new words emerge)
- High language variability - e.g. not standardized like historical languages and dialects
- Polysemy

Tokenizing and Context

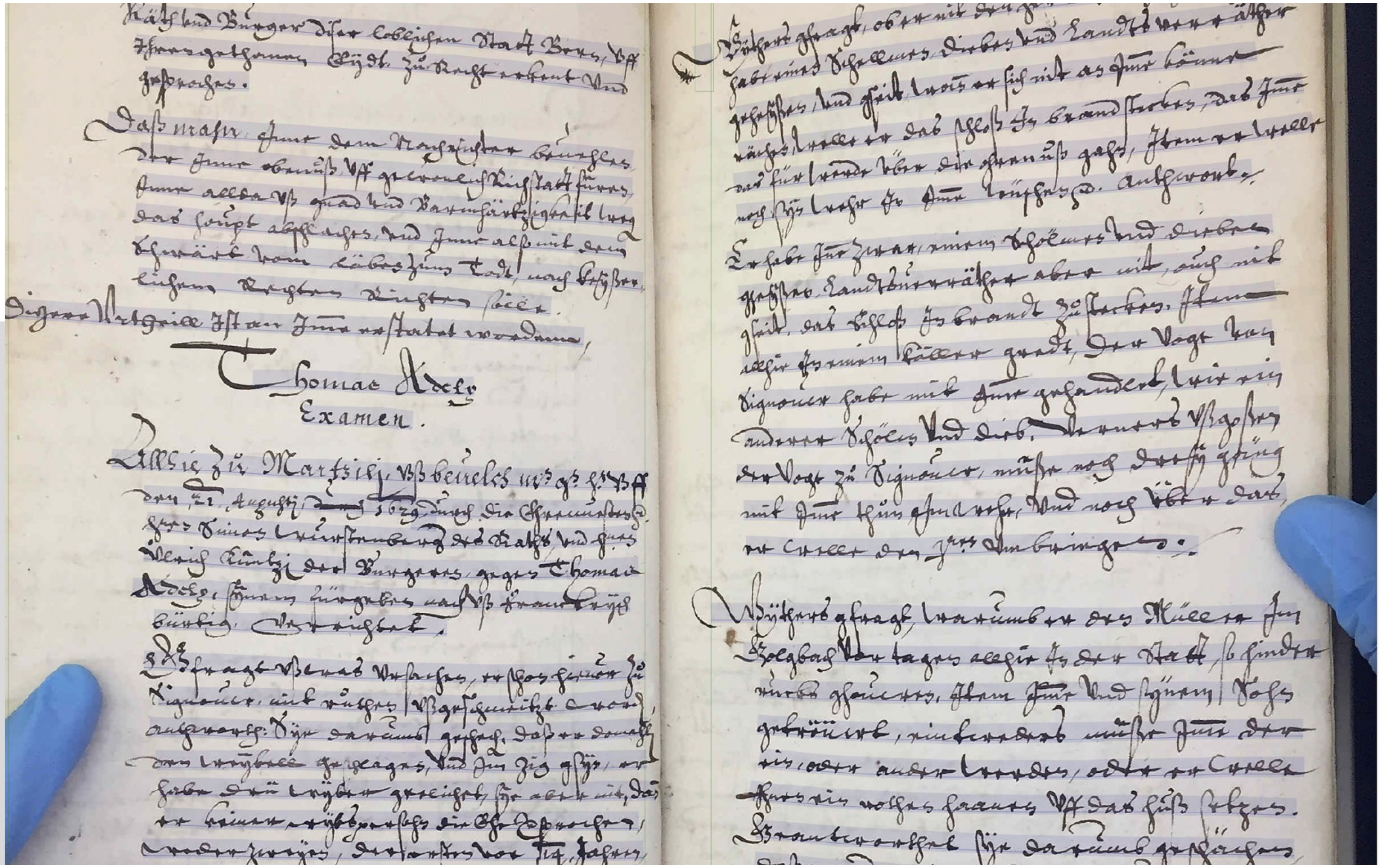
- **Character-based Embeddings:** Similar letter sequences (character-strings) result in similar embeddings.
  (Polysemy not considered.)
- **Sub-Word-Token-Context-Models:** Word embeddings, which vectorize in context (like BPE).
  But: Not working with non-normalized languages!
  - **BERT (sub-word level)**
  - **GPT**
- **Context and character-based models:**
  - **FlairEmbeddings** (https://github.com/flairNLP/flair)
  - **CharacterBert** (https://arxiv.org/abs/2010.10392)

Automated tagging

- As soon as there is a language model to embed your texts, you can train **taggers**.
- Tagger can mark and index entities

- Often used types of tags/tagger in NLP:
    - **Named Entity Recognition (NER-tagging)**
    - **Part-of-Speech (PoS-tagging)**

Named Entity Recognition



2-2 ihrer gethanen Eydt zu recht erkent und

2-3 gesprochen.

2-4 daß mahn, inne dem nachrichter benehlen

2-5 der inne obenuß uff gewarlich kichstatt füren,

2-6 Inne allda uß gnad und Barmhartsigkeit wegen

2-7 das haupt abschlachen, und inne also mit dem

2-8 schwärt vom läben zum Todt nach keyßer-

2-9 lichem Rechten Richten soile.

2-10 digere urtheill ist an ime erstatet worden.

2-11 Thomas Adlis

2-12 Examen.

2-13 Allsie zu Martsili uß benelch gnädigen Herren u

2-14

2-15 den 22. Augusti 1629 durch die ehrenwetsten e

2-16

Named Entity Recognition

- Identification of key information of a text (semantics)
- Classification of entities in a set of predefined categories

George Washington PERSON (February 22, 1732 - December 14, 1799 DATE) was an American NORP soldier, statesman, and Founding Father who served as the first ORDINAL president of the United States GPE from 1789 to 1797 DATE . Appointed by the Continental Congress ORG as commander of the Continental Army ORG , Washington PERSON led the Patriot ORG forces to victory in the American Revolutionary War EVENT , and presided at the Constitutional Convention of 1787 EVENT , which established the Constitution of the United States LAW and a federal government.

Possible categories

- **Person PER**
- **Location LOC**
- **Organization ORG**

Data enrichment in preprocessing → tagging manually
Feature extraction in preprocessing → extract entities with pretrained tagger

Get your text tagged

- Use **spaCy** (all-in-one), **flairNLP** (easy to use), **Transformers** (new superstar)
- Lot’s of pretrained models on **huggingface.co**

- **Preprocessing:**
  - Lowercasing should be avoided, casing is a strong feature of semantics
  - Stopwords may be part of Named Entities (e.g. “of”)
  - Modern systems usually require very little preprocessing

Get the tagged data

- **Lots of data needed for tagger training!**
- **Where to get the labeled data?**
  - DIY
  - Get some help (students, friends, working group)
  - Outsource (pay for it)
  - Crowdsourcing – how to monitor it?
- **Historical data:**
  E.g. “Robert von Habsburg” is a name but also has a place in it.

Part-of-speech tagging

- Recognition of the **syntactic** word type of each token
- E.g. countries = Noun,
  United = Proper Noun
- PoS-tagger works contextualized

```
('The', 'DET')
('representatives', 'NOUN')
('of', 'ADP')
('many', 'ADJ')
('countries', 'NOUN')
('met', 'VERB')
('at', 'ADP')
('the', 'DET')
('United', 'PROPN')
('Nations', 'PROPN')
('conference', 'NOUN')
('in', 'ADP')
('New', 'PROPN')
('York', 'PROPN')
('.', 'PUNCT')
```

Part-of-speech tagging

- Lots of (language specific) **tagsets** available
- spaCy uses **Universal POS tags** https://universaldependencies.org/u/pos
- flairNLP also with multilingulingual taggers https://flairnlp.github.io/docs/intro

- ADJ: adjective
- ADP: adposition
- ADV: adverb
- AUX: auxiliary
- CCONJ: coordinating conjunction
- DET: determiner
- INTJ: interjection
- NOUN: noun
- NUM: numeral
- PART: particle
- PRON: pronoun
- PROPN: proper noun
- PUNCT: punctuation
- SCONJ: subordinating conjunction
- SYM: symbol
- VERB: verb
- X: other