Use Python for NLP

Session 6

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

NLP Frameworks

They provide lots of classes and methods of contemporary…

…**algorithms** for model training and inference

…common **model architectures**

…**tokenizers**

…**visualization** tools

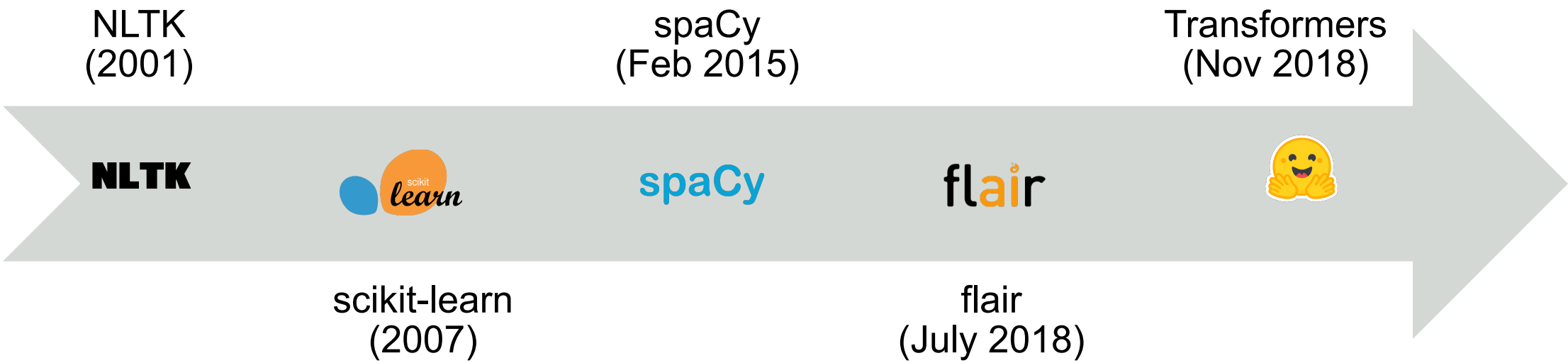
…taggers, lemmatizers, normalizer, and other **preprocessing tools**

…datasets, stopword lists, and **pretrained models**

…**evaluation** tools and measures

Mostly **free and open source.**

Frameworks Overview and Journey



Frameworks Overview and Journey

- **Natural Language Toolkit (NLTK)** *v3.8.1 (January 2023)*
  → nowadays rarely in use, but still present in tutorials
- **Scikit-Learn (sklearn)** *v1.4.0 (January 2024)*
  → easy API, good documentation, and lots of examples
- **spaCy** *v3.7.3 (Feb 2024)*
  → easy API, good visualization
- **flair** *v0.13.1 (December 2023)*
  → Stackable and character embeddings, good German support
- **Transformers by Huggingface** *v4.37.2 (January 2024)*
  → Powerful architectures, Huggingface platform for pretrained models and datasets, fast integration of latest developments

spaCy

spaCy

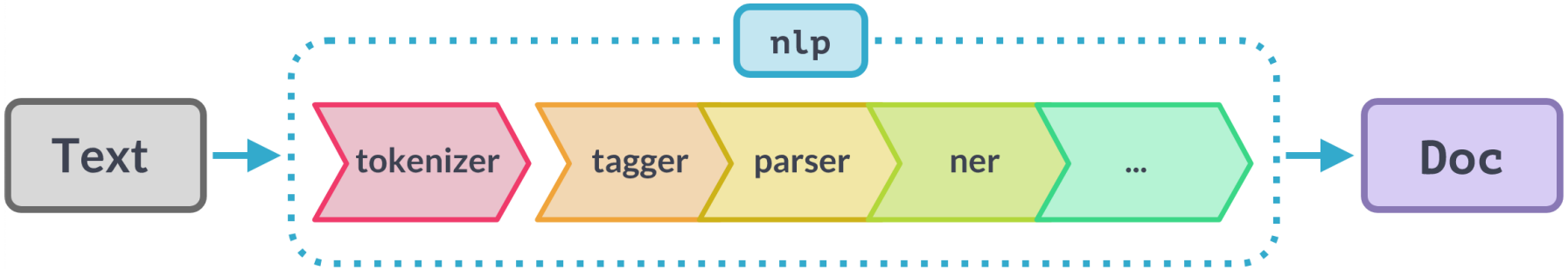
- **Fast** and **easy to use**
  "industrial-strength natural language processing"
- **Pipeline** based API → feeding text to a model-pipeline
- **Convolutional-Neural-Network (CNN)** and **transformers** architectures
- **84 pretrained pipelines for 25 languages**
- Built in **visualizers** for syntax and NER

https://spacy.io

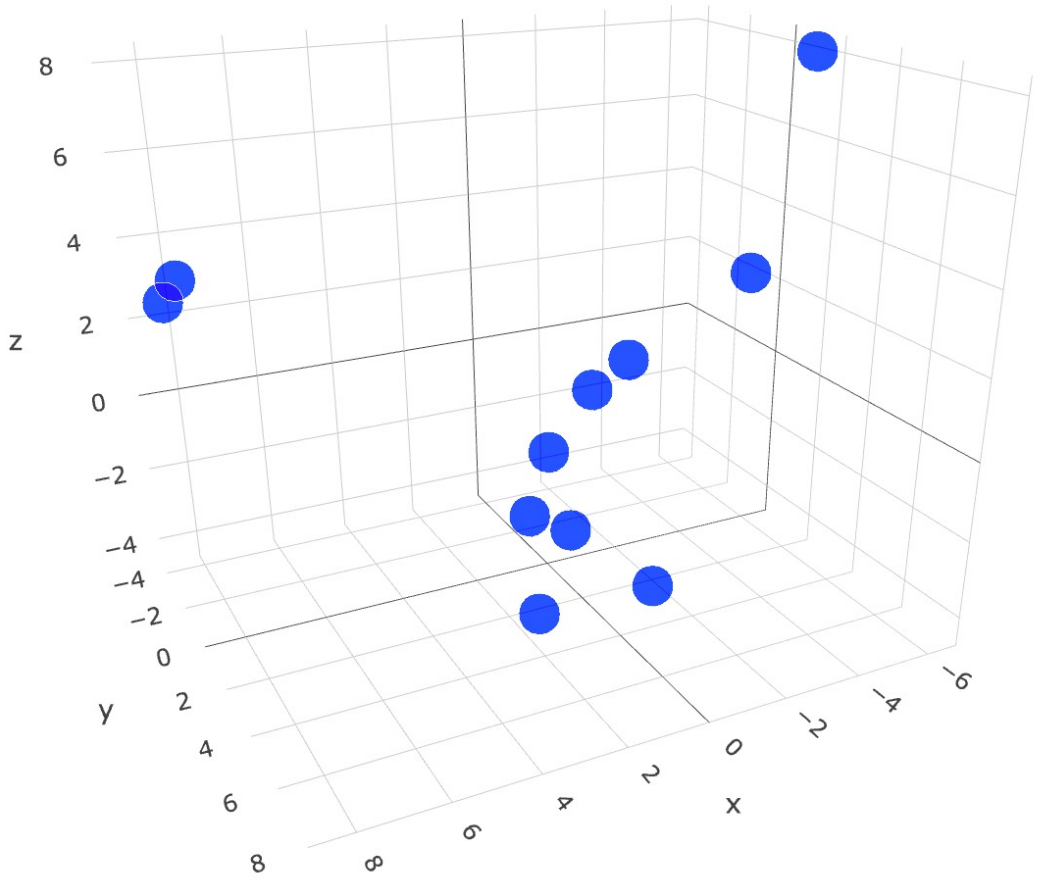
spaCy pipeline

- Load a model from: https://spacy.io/models
- "efficiency" = CNN (`en_core_web_sm`)
  "accuracy" = Transformers (`en_core_web_trf`)
- Process your text with the `nlp()`-Method

- Get Tokens with attributes like:
  - `token.text`
  - `token.lemma`
  - `token.is_punct`
  - `token.pos`
  - `token.ent_type`
  - `token.sentiment`
  - `token.is_stop`



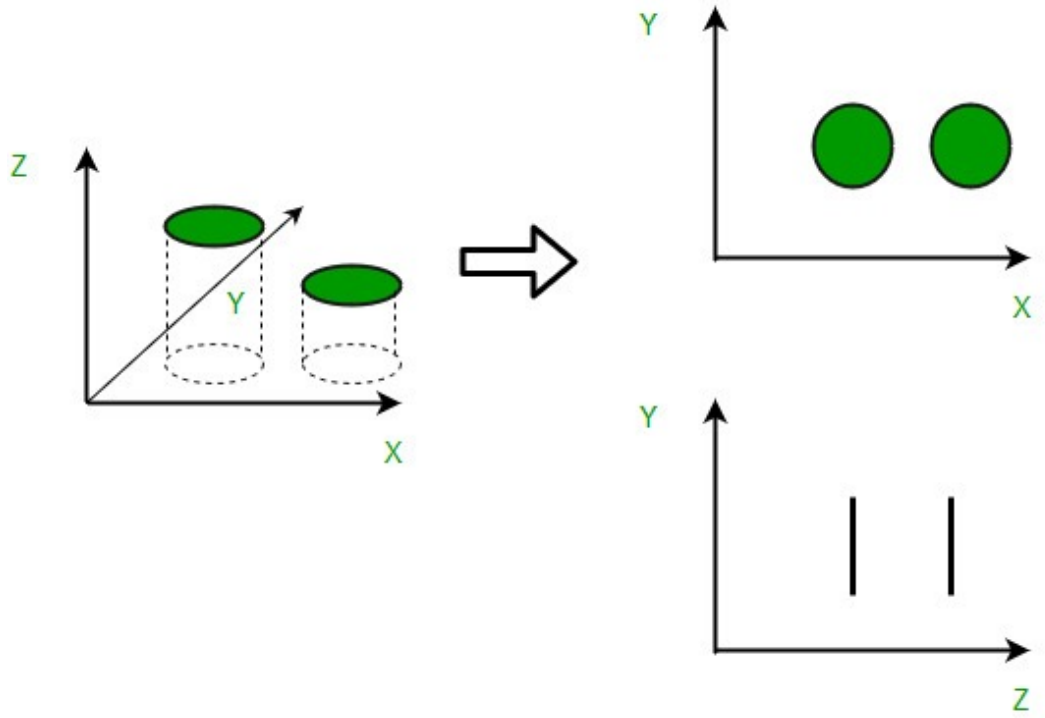
Visualization



Imagine

```
bertEmbedding = TransformerWordEmbeddings('bert-large-cased’)
sentence = Sentence('She puts her arm around on his shoulder.’)
bertEmbedding.embed(sentence)
```

→ **Tensor with 1024 dimensions.**



Pros/Cons of Dimensionality Reduction

Pros

- Visualize what you're working with
- Data compression and reduced storage space
- Reduces computation time

Cons

- Some information in the data is lost
- Understanding the assumptions of the algorithm

Polysemy

Words with multiple meanings → Semantic context

- “She sat on the **arm** of the sofa.”
- “It was a bad idea to **arm** the bomb.”
- “Joe grabbed Bob's **arm**.”

Embed with embeddings which understands the context of a word. → E.g. BERT

Demo Embeddings Glove and BERT

- **Polysemy “arm” / “sanctus”**
- Dimensional reduction with **Principal Component Analysis (PCA)**