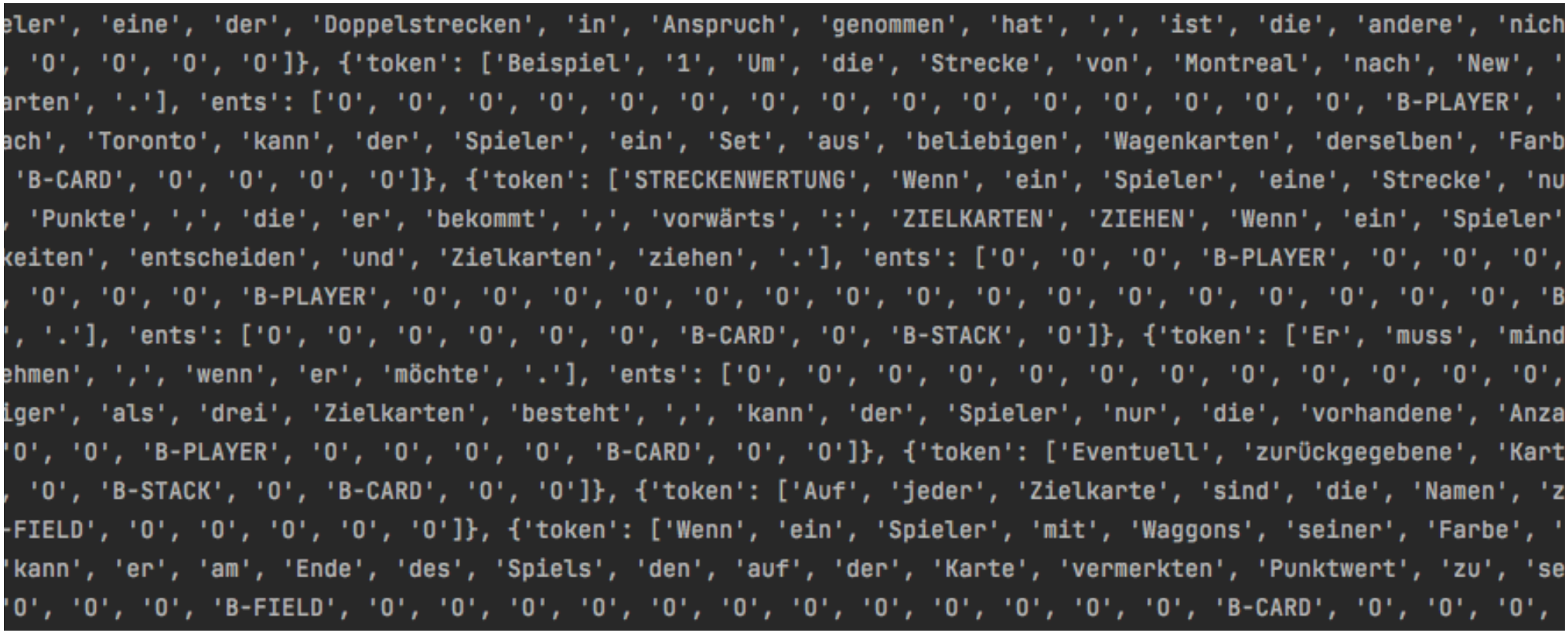
TECHNISCHE HOCHSCHULE NÜRNBERG
GEORG SIMON OHM



Tutorial: How to train a NER model from annotion to model with Spacy V3 and Inception Plattform

Alexander Schneider

1. July 2021

Tutorial

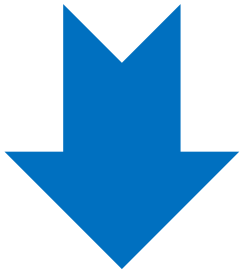
- Slides as fallback and for memory support
- Step by step
- Goal: Everything to know about
  - to start a own project
  - find the required resources

Agenda

- Text Corpus

The source to start work with

- Setup Inception Project
- Annotation
- Export



Annotation process with Inception [1]

- Setup Spacy Pipeline
- Training
- Evaluation

Train NER model with Spacy v3 [2]

- Application

The destination to go to!

Text Corpus

- Example text from Kaggle
- Star Trek Scripts
  - Raw texts scripts of all Star Trek series
  - https://www.kaggle.com/gjbroughton/start-trek-scripts
- Extracted Text:
  - Commander Data
  - First 8 episodes of „Next Generation“
- Only used for demonstration of Inception [1]

Setup Inception Project

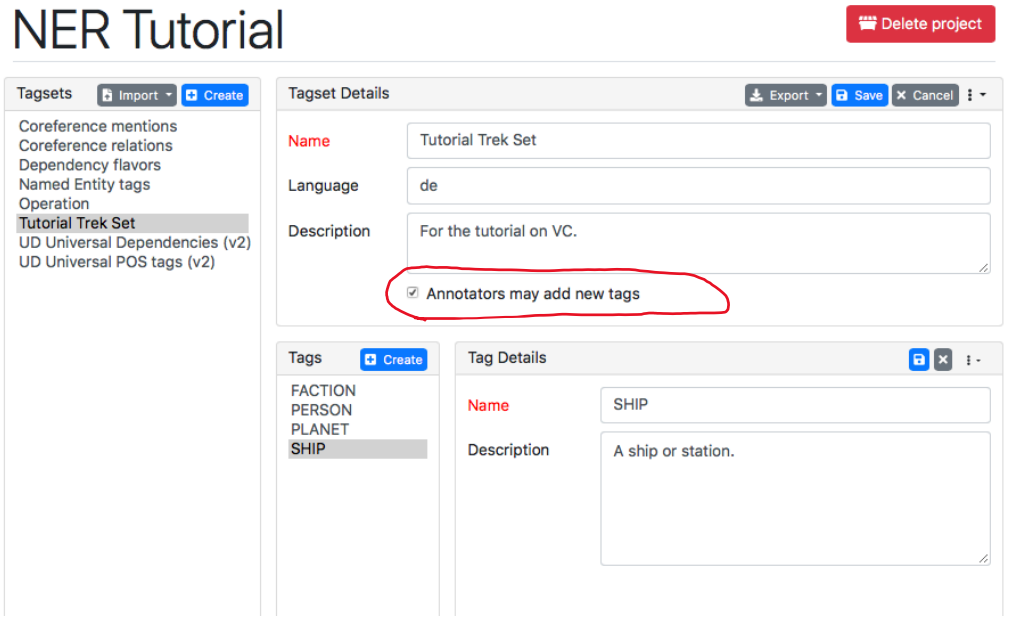
- *Install Inception[1] Instance (see also A1)*

- Create a Project and
  - add your Tagset for Named Entities
  - add Layer and Features
  - add Documents
  - add Recommender (optional)
- Annotation Phase
- Export

Add Tagset

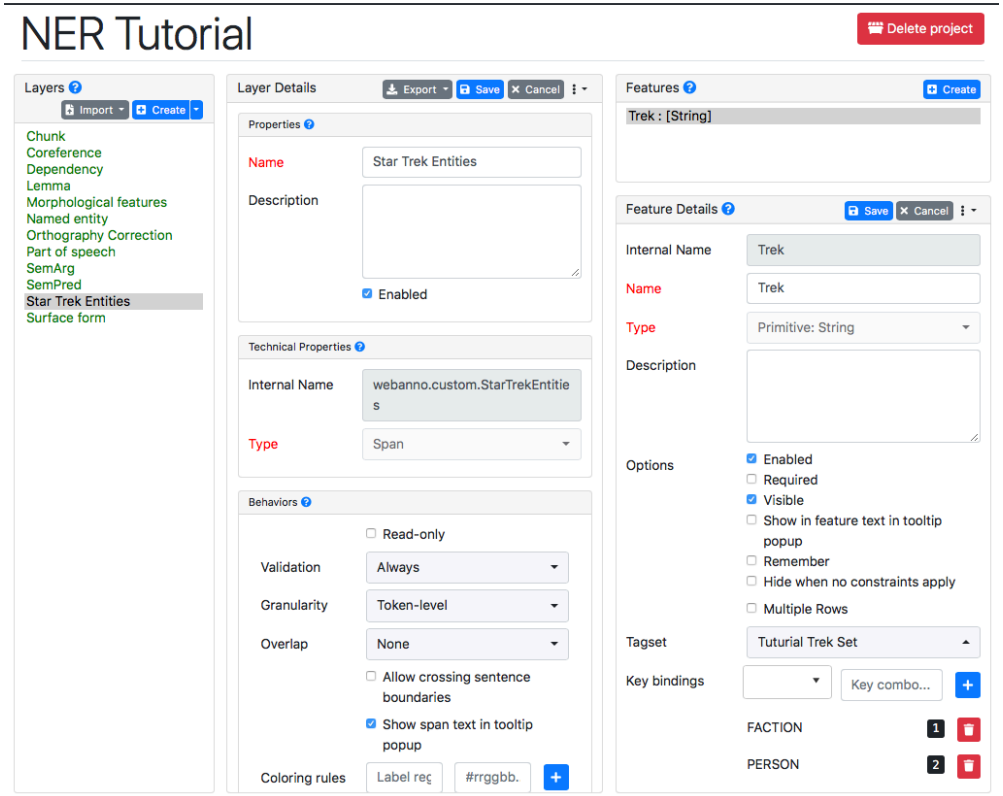
- Chose wisely to enable create
- Add entry for every entity
- Create a set for every type
- You can add more Entities later

Rename a Tag later is *funny*…
No renaming of already created annotations!



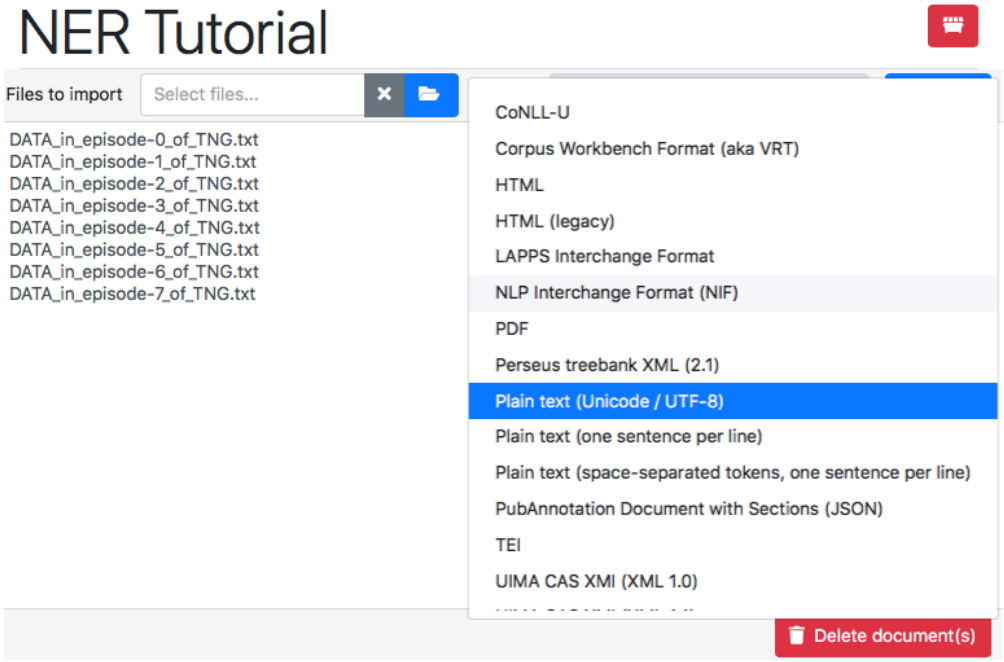
Add Layer

- Chose name and behaviors
- Add Features carrying values
- Key binding
  - Nice feature to work faster
  - Select entity by keyboard
  - Or one click



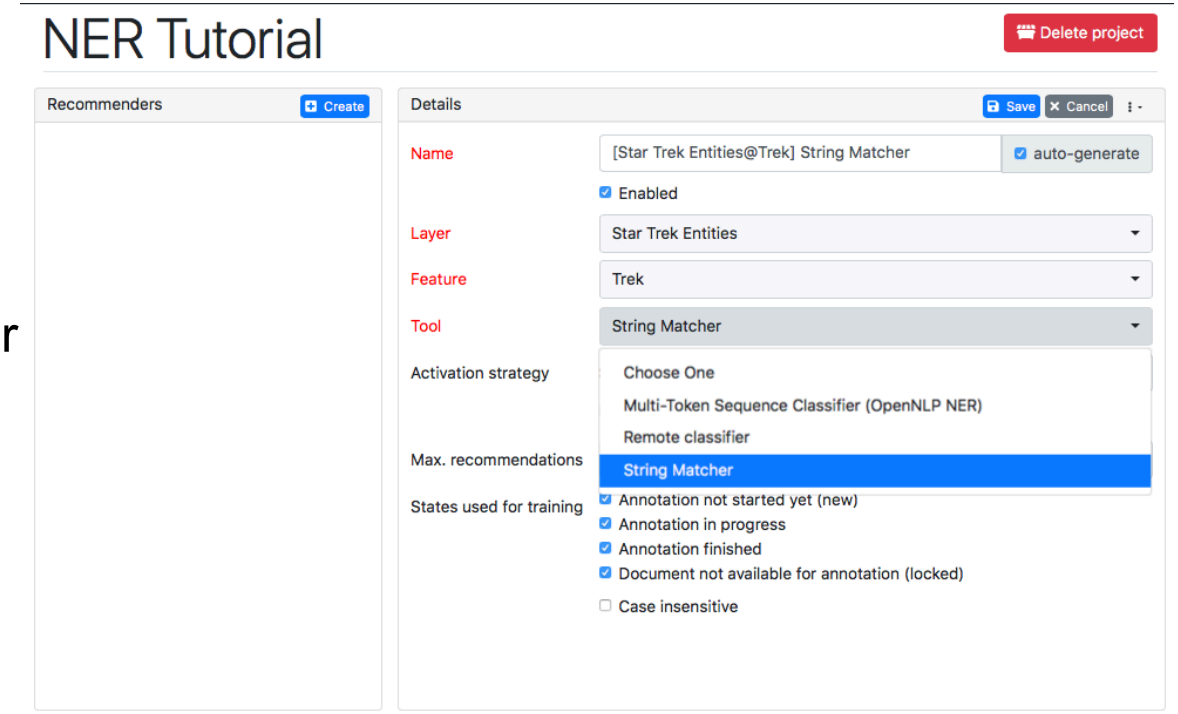
Add Documents

- Add your documents to project
- You can later
  - add more
  - remove again
  - but not change…
- Different formats are supported
  - Text and PDFs works fine
  - Nice if corpus already exists

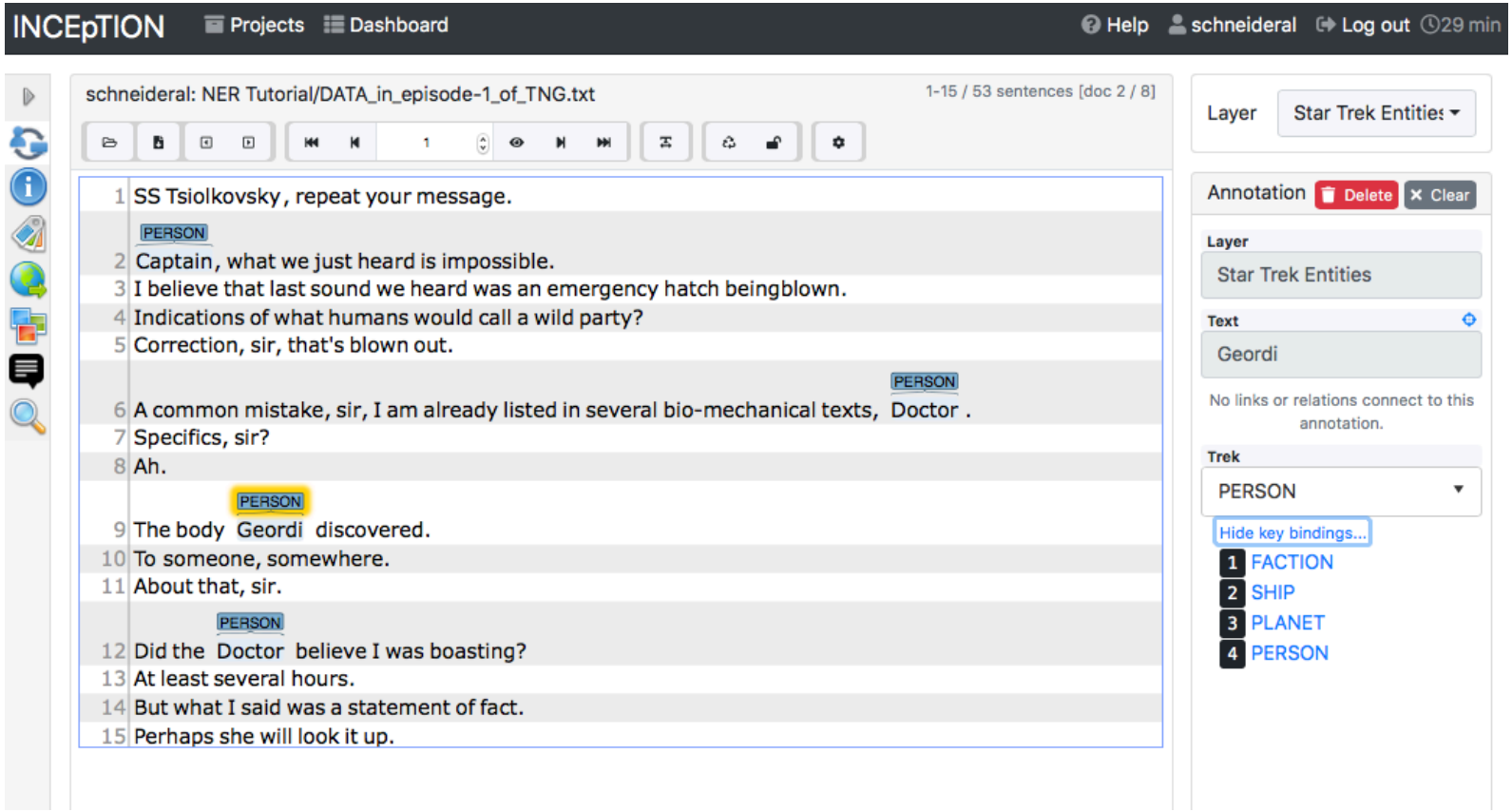


Add Recommender (optional)

- String Matcher!
  - Out of the box helpful
- Remote classifier enable to implement own recommender [6]
- Not required, but very useful



Annotation Phase



Export

- Different Formats available
- For easy post-processing
  - WebAnno TSV v3.2
  - Tab separated text format
  - Well documented

#FORMAT=WebAnno TSV 3.3			
#T_SP=webanno.custom.StarTrekEntities|Trek			
#Text=Simply solve the mystery of Farpoint Station.			
2-1	11-17	Simply	_
2-2	18-23	solve	_
2-3	24-27	the	_
2-4	28-35	mystery	_
2-5	36-38	of	_
2-6	39-47	Farpoint	SHIP[1]
2-7	48-55	Station	SHIP[1]
2-8	55-56	.	_

Train NER model

- Step by Step
  - Setup Spacy Pipeline
  - Pre-process input documents
  - Train model
  - Evaluation

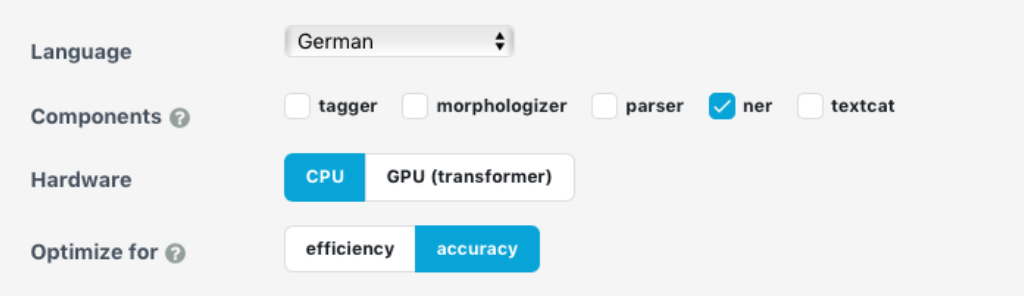
- Spacy Project
  - Everything together

Setup Spacy Pipeline

- Pipelines are recommended way for training
  - But programmatically is possible, too
- Create a config file with all settings

```
spacy init config config.cfg --lang de --pipeline ner --optimize accuracy
```

- Or use quick start widget



https://spacy.io/usage/training

Setup Spacy Pipeline

- All configuration are made here
- Single point of configuration
- All possible parameters are defined
  - Even the not used
  - Prevent strange behavior if defaults change in future version
- No need for looooonnng command lines
- Documentation and Versioning

```
[corpora.train]
@readers = "spacy.Corpus.v1"
path = ${paths.train}
max_length = 2000
gold_preproc = true
limit = 0
augmenter = null

[training]
train_corpus = "corpora.train"
dev_corpus = "corpora.dev"
seed = ${system.seed}
gpu_allocator = ${system.gpu_allocator}
dropout = 0.1
accumulate_gradient = 1
patience = 1600
max_epochs = 0
```

Example excerpt of config.cfg

Pre-process input documents

- Spacy 3 requires IOB-Tags [3]
  - some converter included [4]
- Transform TSV -> IOB required
- Python scripting required
  - e.g. [7] lack support for multiple features

FIELD[4]

[(', 'Das', 'sind', ':', 'País', 'Vasco', ',', 'Aragón', 'und', 'Valencia', ',', 'nicht', 'jedoch', 'das', 'Castillo', ')']

['O', 'O', 'O', 'O', 'B-FIELD', 'I-FIELD', 'O', 'B-FIELD', 'O', 'B-FIELD', 'O', 'O', 'O', 'O', 'O', 'O', 'O']

B	Begin
I	Inside
O	Outside

```
doc_bin = DocBin()
for current in data:
    doc = Doc(vocab, words=current['token'], ents=current['ents'])
    doc_bin.add(doc)
```

Pre-process input documents

- Create 2 files with IOB-Data from annotation files
  - Training Data (train.spacy)
  - Validation Data (dev.spacy)
- Check data before training

```
spacy debug data configs/config.cfg \
       --paths.train corpus/train.spacy \
       --paths.dev corpus/dev.spacy
```

Pre-process input documents

- Output of actual project
- Analyze output and decide to: go back to annotation or start training

```
============================== Training stats ==============================
Language: de
Training pipeline: tok2vec, ner
5823 training docs
1456 evaluation docs
⚠ 41 training examples also in evaluation data

============================== Vocab & Vectors ==============================
ℹ 106343 total word(s) in the data (9302 unique)
ℹ No word vectors present in the package

========================== Named Entity Recognition ==========================
ℹ 14 label(s)
0 missing value(s) (tokens with '-' label)
⚠ 2 entity span(s) with punctuation
⚠ Low number of examples for label 'PEN' (10)
⚠ Low number of examples for label 'BAG' (30)
⚠ Low number of examples for label 'BOX' (50)
⚠ Low number of examples for label 'CLOCK' (5)
⚠ Low number of examples for label 'INSTRUCTION' (25)
✓ Examples without occurrences available for all labels
✓ No entities consisting of or starting/ending with whitespace
```

Train NER Model

- Start Training!

```
spacy train configs/config.cfg --output trainig/ --paths.train corpus/train.spacy \
        --paths.dev corpus/dev.spacy
```

```
============================ Initializing pipeline ============================
[2021-06-30 12:54:11,495] [INFO] Set up nlp object from config
[2021-06-30 12:54:11,513] [INFO] Pipeline: ['tok2vec', 'ner']
[2021-06-30 12:54:11,520] [INFO] Created vocabulary
[2021-06-30 12:54:11,520] [INFO] Finished initializing nlp object
[2021-06-30 12:54:19,530] [INFO] Initialized pipeline components: ['tok2vec', 'ner']
✓ Initialized pipeline
============================= Training pipeline =============================
ℹ Pipeline: ['tok2vec', 'ner']
ℹ Initial learn rate: 0.0
E    #       LOSS TOK2VEC  LOSS NER  ENTS_F  ENTS_P  ENTS_R  SCORE
---  ------  ------------  --------  ------  ------  ------  ------
  0       0          0.00     52.48    0.00    0.00    0.00    0.00
  0     600         87.50   3795.58   22.46   56.12   14.04    0.22
  1     800        142.61   4123.08   22.93   37.23   16.57    0.23
```

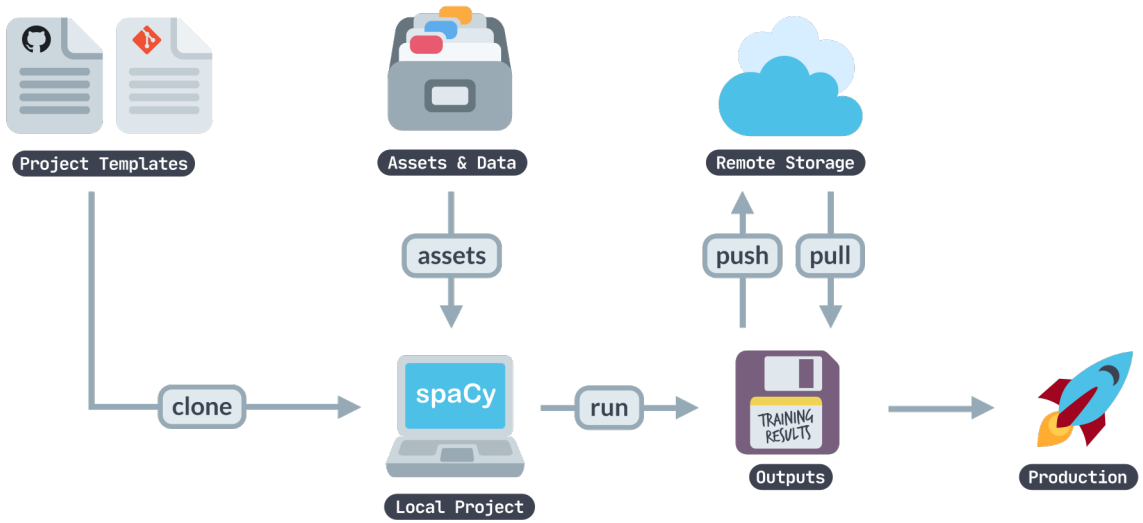
Evaluation

```
spacy evaluate training/model-best corpus/dev.spacy --output training/metrics.json
```

```
================================== Results ==================================

TOK     -
NER P   88.14
NER R   91.60
NER F   89.84
SPEED   7391

============================== NER (per type) ==============================

                P        R        F
CARD        92.04    91.36    91.70
TOKEN       89.62    95.35    92.39
PLAYER      93.30    96.91    95.07
FIELD       80.32    86.17    83.14
BOX         92.86   100.00    96.30
BOARD       94.55    88.14    91.23
TILE        87.54    88.15    87.85
FIGURE      85.33    93.57    89.26
DICE       100.00   100.00   100.00
STACK       73.47    85.71    79.12
INSTRUCTION 100.00    63.64    77.78
PEN           0.00     0.00     0.00
BAG         100.00   100.00   100.00

✓ Saved results to training/metrics.json
```

Spacy Projects

- Put everything together
- End-to-End workflows
- Reproduce results with complete execution pipeline and data
- Versioning (e.g. Git) of results for history or publication



https://spacy.io/usage/projects

Spacy Projects

- Clone or download project template [5]
- Exchange "`config.cfg`" by own NLP Spacy pipeline
- Customize processing workflow
  - Defined in `project.yml`

```
- name: "train"
  help: "Train a named entity recognition model"
  script:
    - "python -m spacy train configs/${vars.config} --output training/
       --paths.train corpus/${vars.train}.spacy --paths.dev corpus/${vars.dev}.spacy"
  deps:
    - "corpus/${vars.train}.spacy"
    - "corpus/${vars.dev}.spacy"
  outputs:
    - "training/model-best"
```

Spacy Projects

- Execute a workflow step

```
spacy project run train
```

- Execute all at once
  - Create workflow definition in “`project.yml`”

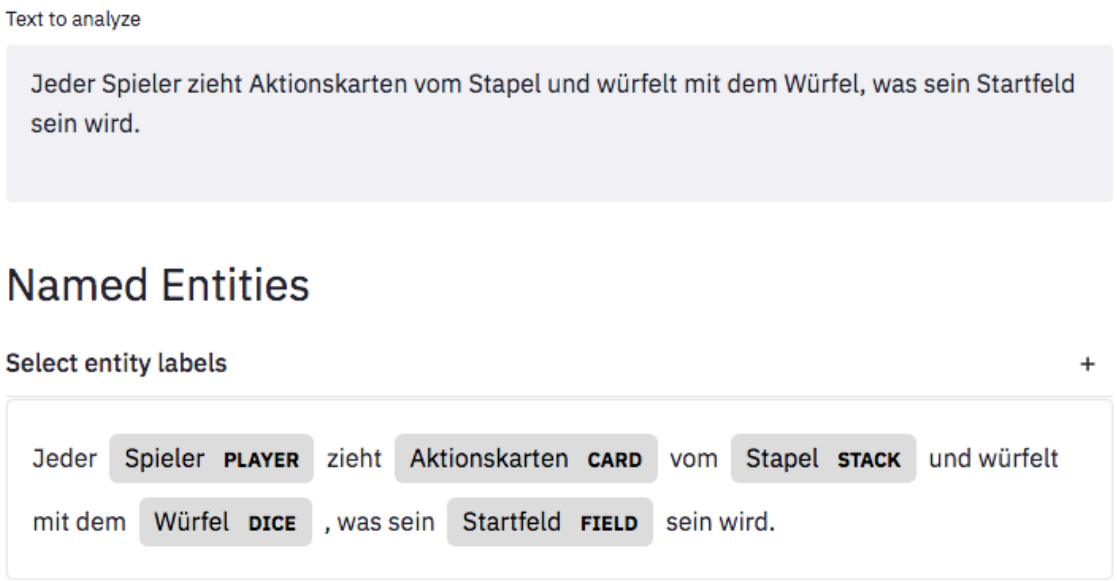
```
workflows:
  all:
    - preprocess
    - train
    - evaluate
```

  - Execute: `spacy project run all`

Application

- Apply model to text
- Using streamlit for interactive UI

```
spacy_streamlit.visualize(models, default_text, visualizers=["ner"])
```



Referenzen

[1] Klie, J.-C., Bugert, M., Boullosa, B., Eckart de Castilho, R. and Gurevych, I. (2018): The INCEpTION Platform: Machine-Assisted and Knowledge-Oriented Interactive Annotation. In Proceedings of System Demonstrations of the 27th International Conference on Computational Linguistics (COLING 2018), Santa Fe, New Mexico, USA (https://inception-project.github.io)

[2] Spacy, Natural Language Processing, https://spacy.io

[3] Ramshaw, Marcus (1995): Text Chunking using Transformation-Based Learning, (https://arxiv.org/abs/cmp-lg/9505040)

[4] Spacy, converter for converting into Spacy binary format, https://spacy.io/api/cli#convert

[5] Project Templates for Spacy, https://github.com/explosion/projects/tree/v3/pipelines

[6] Tutorial Custom Recomender for Inception Plattfrom, https://github.com/inception-project/inception-external-recommender/blob/master/Tutorial.ipynb

[7] web-anno-tsv 0.0.1, Library for parsing WebAnno TSV, https://pypi.org/project/web-anno-tsv/

[8] streamlit.io, Convert data scripts into shareable web apps, https://streamlit.io

A1: Inception Docker Compose

- Paste following code into a file named “docker-compose.yml”
- Excecute: docker-compose up
- Open Inception: http://localhost:8080

```
version: '3.7'
services:
  mysqlserver:
    image: "mysql:5"
    environment:
      - MYSQL_RANDOM_ROOT_PASSWORD=yes
      - MYSQL_DATABASE=inception
      - MYSQL_USER=tutorial
      - MYSQL_PORT=3306
      - MYSQL_PASSWORD=annodb
    volumes:
      - mysql_data:/var/lib/mysql
    command: ["--character-set-server=utf8", "--collation-server=utf8_bin"]
```

A1: Inception Docker Compose

```
  webserver:
    image: inceptionproject/inception:0.19.7
    ports:
      - 8080:8080
    environment:
      - INCEPTION_DB_DIALECT=org.hibernate.dialect.MySQL5InnoDBDialect
      - INCEPTION_DB_DRIVER=com.mysql.jdbc.Driver
      - INCEPTION_DB_URL=jdbc:mysql://mysqlserver:3306/inception?useUnicode=true&characterEncoding=UTF-8
      - INCEPTION_DB_USERNAME=tutorial
      - INCEPTION_DB_PASSWORD=annodb
    volumes:
      - inception_data:/export
volumes:
  inception_data:
  mysql_data:
```