



spaCy and the future of multi-lingual NLP

Matthew Honnibal

Ines Montani

Explosion



Matthew Honnibal

CO-FOUNDER

PhD in Computer Science in 2009.
10 years publishing research on state-of-the-art natural language understanding systems.
Left academia in 2014 to develop spaCy.



Ines Montani

CO-FOUNDER

Programmer and front-end developer with degree in media science and linguistics.
Has been working on spaCy since its first release. Lead developer of Prodigy.

Early 2015

spaCy is first released

- open-source library for industrial-strength **Natural Language Processing**
- focused on **production use**



Early 2015

spaCy is first released

- open-source library for industrial-strength **Natural Language Processing**
- focused on **production use**

Current stats

100k+ users worldwide

15k stars on GitHub

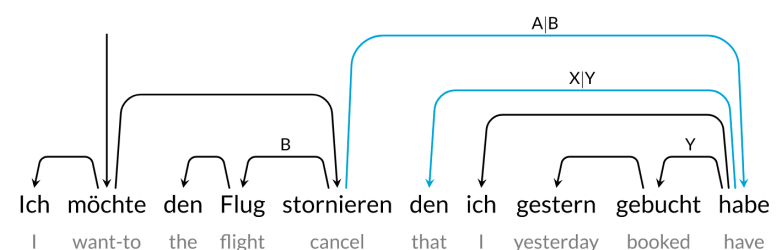
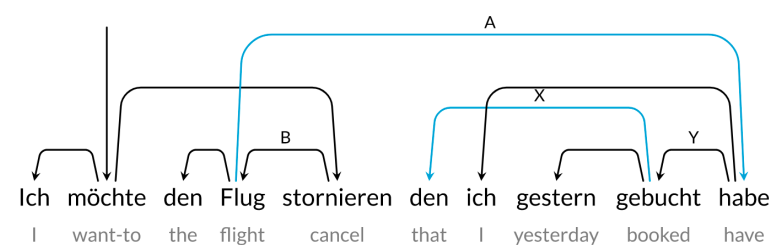
400 contributors

60+ extension packages



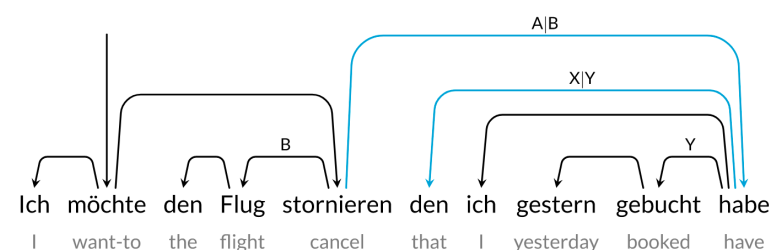
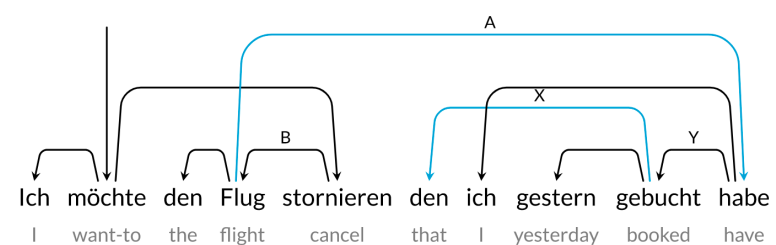
Early 2016

German model



Early 2016

German model



Current stats


52+ supported languages

23 pre-trained statistical models
for **21** languages

A horizontal dotted line on the left side of the slide.

Late 2016


Explosion

- new company for AI **developer tools**
 - **bootstrapped** through consulting for the first 6 months
 - funded through **software sales** since 2017
 - **remote** team, centered in **Berlin**
- 
- A large, sweeping dotted arc that curves from the bottom left towards the bottom right of the slide.

A horizontal dotted line on the left side of the slide.

Late 2016

Explosion

- new company for AI **developer tools**
 - **bootstrapped** through consulting for the first 6 months
 - funded through **software sales** since 2017
 - **remote** team, centered in **Berlin**
- 
- A curved dotted line starting from the bottom left and ending near the right side of the slide.

Current stats

7 team members

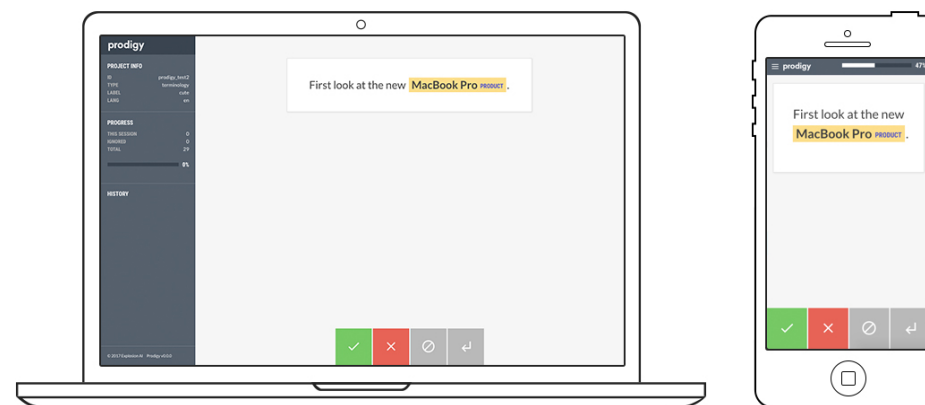
100% independent & profitable

A curved dotted line starting from the bottom right and ending near the right side of the slide.

Late 2017

Prodigy

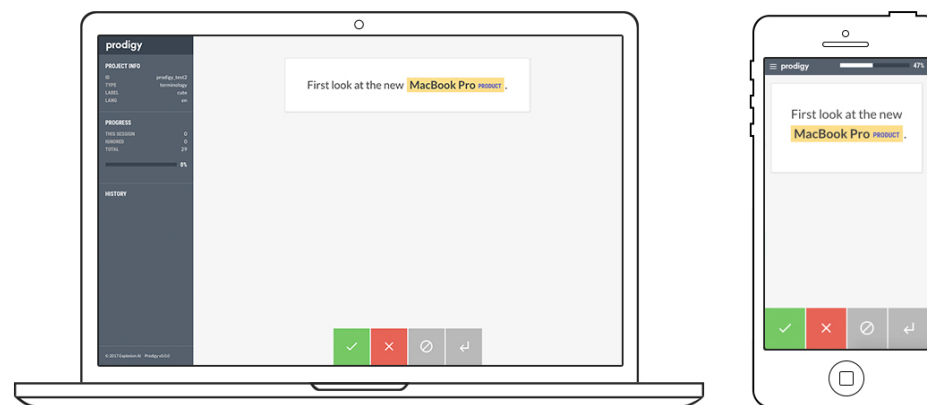
- first commercial product
- modern **annotation tool**
- fully scriptable in Python



Late 2017

Prodigy

- first commercial product
- modern **annotation tool**
- fully scriptable in Python



Current stats

2500+ users, including

250+ companies

1200+ forum members

**Is NLP becoming
more or less
multi-lingual?**

The good



The good



Universal Dependencies

- **100+** treebanks, **70+** languages, **1** annotation scheme
- driving lots of new **multi-lingual parsing** research

The good



Universal Dependencies

- **100+** treebanks, **70+** languages, **1** annotation scheme
- driving lots of new **multi-lingual parsing** research

More work in the field

- huge increase in the number of papers on **all NLP topics** – including multi-lingual
- lots of crossover from **general ML** work, too

The good



Universal Dependencies

- **100+** treebanks, **70+** languages, **1** annotation scheme
- driving lots of new **multi-lingual parsing** research

More work in the field

- huge increase in the number of papers on **all NLP topics** – including multi-lingual
- lots of crossover from **general ML** work, too

Transfer Learning

- we're much better at learning from **unlabeled text** (e.g. Wikipedia)
- leverage resources so we need **less annotation** per language

The bad



The bad



More competitive

- “winning” a shared task could be worth millions (**fame** and **future salary**)
- few researchers care about the languages

The bad

More competitive

- “winning” a shared task could be worth millions (**fame** and **future salary**)
- few researchers care about the languages

More expensive

- experiments now cost a lot to run (especially **GPU**)
- results are **unpredictable**
- pressure to run **fewer** experiments on **fewer** datasets

The bad



More competitive

- “winning” a shared task could be worth millions (**fame** and **future salary**)
- few researchers care about the languages

More expensive

- experiments now cost a lot to run (especially **GPU**)
- results are **unpredictable**
- pressure to run **fewer** experiments on **fewer** datasets

Faster moving, less careful

- huge pressure to **publish**
- volume of publications makes reviewing more random
- dynamics promote **incremental work**

What we need: annotated data



What we need: annotated data

Annotated carefully

ideally by small teams of
experts

What we need: annotated data

Annotated carefully

ideally by small teams of
experts

Extensive experiments

designed to answer questions
not optimize benchmarks

What we need: annotated data

Annotated carefully

ideally by small teams of
experts

Extensive experiments

designed to answer questions
not optimize benchmarks

Maintained datasets

not just static resources that
never improve

Prodigy Annotation Tool

<https://prodi.gy>



The screenshot shows the Prodigy Annotation Tool interface. On the left is a sidebar with three sections: "PROJECT INFO" (showing PROJECT_ID and VIEW_ID), "PROGRESS" (showing THIS SESSION and TOTAL counts with a 0% progress bar), and "HISTORY" (showing recent annotations with status icons). The main area displays a sentence: "This guy built a travel dock for his Nintendo Switch and I want one", with "Nintendo Switch" highlighted as a "PRODUCT" annotation. At the bottom is a control bar with four buttons: a green checkmark, a red X, a grey circle with a slash, and a grey left arrow. Callout boxes point to these elements: "annotation project details" points to the sidebar; "current progress" points to the progress bar; "recent annotations" points to the history list; "focus on one task at a time" points to the main text area; "accept, reject or ignore annotation" points to the first three buttons; and "undo" points to the left arrow button.

annotation project details

current progress

recent annotations

focus on one task at a time

accept, reject or ignore annotation

undo

Prodigy Annotation Tool

<https://prodi.gy>



```

@prodigy.recipe("my-recipe")
def my_recipe_script(dataset, input_file, batch_size=16):
    stream = JSONL(input_file)


    def update(answers):
        print(f"Received {len(answers)} answers")

    return {
        "dataset": dataset,
        "stream": stream,
        "update": update,
        "config": {"batch_size": batch_size}
    }
```



Thank you!

 **Explosion**
explosion.ai

 **Follow us on Twitter**
@honnibal
@_inesmontani
@explosion_ai