# Practical transfer learning for NLP with spaCy and Prodigy

Ines Montani
**Explosion AI**

BERT

Google AI Blog

Open Sourcing BERT: State-of-the-Art Pre-training for Natural Language Processing

Friday, November 2, 2018

Posted by Jacob Devlin and Ming-Wei Chang, Research Scientists, Google AI Language

ELMo

**Deep contextualized word representations**

**Matthew E. Peters**[†], **Mark Neumann**[†], **Mohit Iyyer**[†], **Matt Gardner**[†],
{matthewp,markn,mohiti,mattg}@allenai.org

**Christopher Clark**[*], **Kenton Lee**[*], **Luke Zettlemoyer**[†*]
{csquared,kentonl,lsz}@cs.washington.edu

EXPLOSION

ULMFiT

**Universal Language Model Fine-tuning for Text Classification**

**Jeremy Howard**[*]
fast.ai
University of San Francisco
j@fast.ai

**Sebastian Ruder**[*]
Insight Centre, NUI Galway
Aylien Ltd., Dublin
sebastian@ruder.io

BERT

Google AI Blog

Open Sourcing BERT: State-of-the-Art Pre-training for Natural Language Processing

Friday, November 2, 2018

Posted by Jacob Devlin and Ming-Wei Chang, Research Scientists, Google AI Language

ELMo

**Deep contextualized word representations**

**Matthew E. Peters**[†], **Mark Neumann**[†], **Mohit Iyyer**[†], **Matt Gardner**[†],
{matthewp,markn,mohiti,mattg}@allenai.org

**Christopher Clark**[*], **Kenton Lee**[*], **Luke Zettlemoyer**[†*]
{csquared,kentonl,lsz}@cs.washington.edu

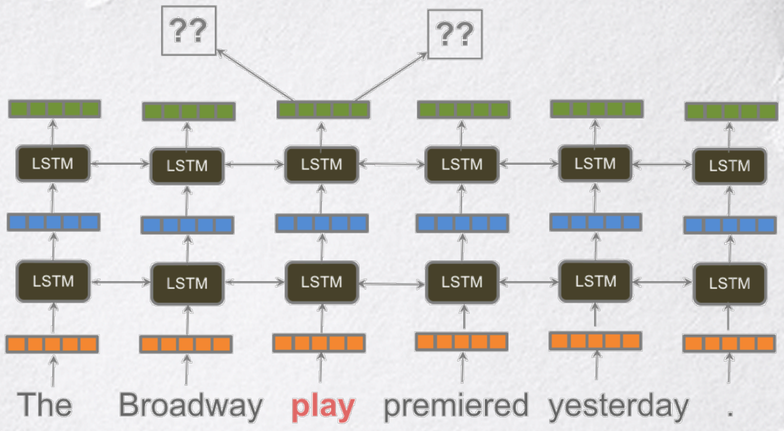
EXPLOSION

ULMFiT

**Universal Language Model Fine-tuning for Text Classification**

**Jeremy Howard**[*]
fast.ai
University of San Francisco
j@fast.ai

**Sebastian Ruder**[*]
Insight Centre, NUI Galway
Aylien Ltd., Dublin
sebastian@ruder.io




NLP's ImageNet moment has arrived

08.JUL.2018

EXPLOSION

BERT

Google AI Blog

Open Sourcing BERT: State-of-the-Art Pre-training for Natural Language Processing

Friday, November 2, 2018

Posted by Jacob Devlin and Ming-Wei Chang, Research Scientists, Google AI Language

ELMo

**Deep contextualized word representations**

**Matthew E. Peters**[†], **Mark Neumann**[†], **Mohit Iyyer**[†], **Matt Gardner**[†],
{matthewp,markn,mohiti,mattg}@allenai.org

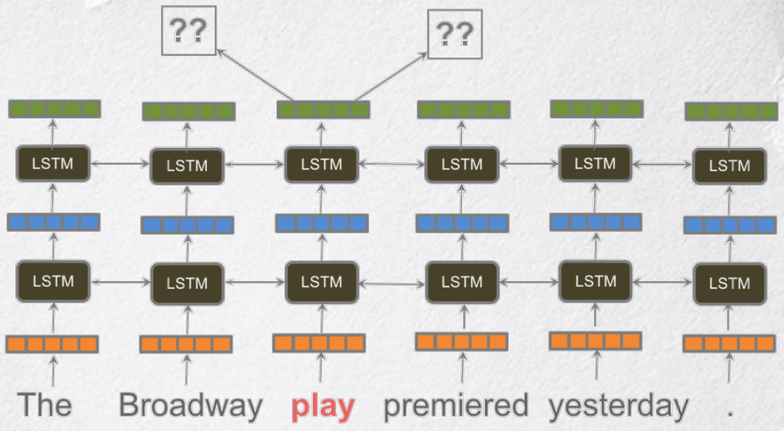
**Christopher Clark**[*], **Kenton Lee**[*], **Luke Zettlemoyer**[†*]
{csquared,kentonl,lsz}@cs.washington.edu

ULMFiT

**Universal Language Model Fine-tuning for Text Classification**

**Jeremy Howard**[*]
fast.ai
University of San Francisco
j@fast.ai

**Sebastian Ruder**[*]
Insight Centre, NUI Galway
Aylien Ltd., Dublin
sebastian@ruder.io




NLP's ImageNet moment has arrived

08.JUL.2018

**Machines learn language better by using a deep understanding of words**

TechCrunch

*Finally, a Machine That Can Finish Your Sentence*

The New York Times

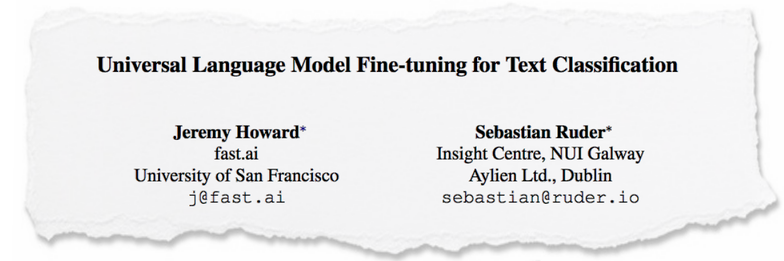
Completing someone else's thought is not an easy trick for A.I. But new systems are starting to crack the code of natural language.

# 💬 Language is more than just words

- NLP has always struggled to get beyond a "bag of words"
- Word2Vec (and GloVe, FastText etc.) let us pretrain **word meanings**
- How do we learn the meanings of words **in context**? Or **whole sentences**?

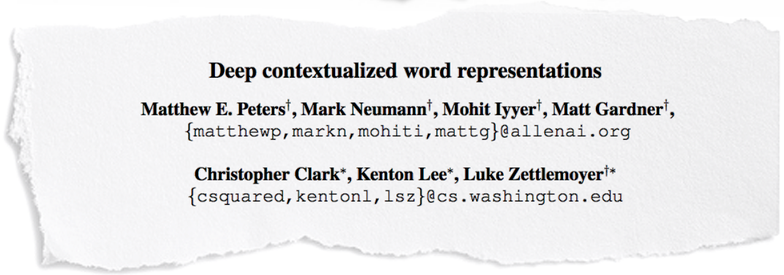
# Language model pretraining

- ULMFiT, ELMo: Predict the **next word** based on the previous words


Universal Language Model Fine-tuning for Text Classification

Jeremy Howard*
fast.ai
University of San Francisco
j@fast.ai

Sebastian Ruder*
Insight Centre, NUI Galway
Aylien Ltd., Dublin
sebastian@ruder.io

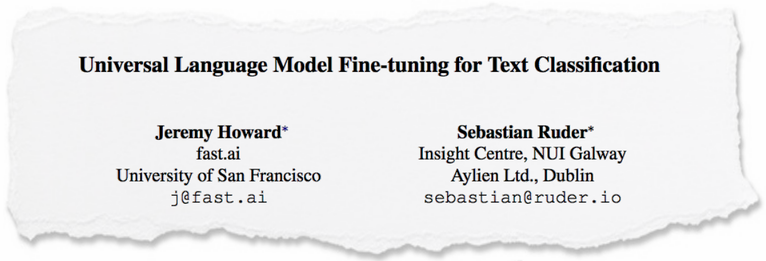

Deep contextualized word representations

Matthew E. Peters†, Mark Neumann†, Mohit Iyyer†, Matt Gardner†,
{matthewp,markn,mohiti,mattg}@allenai.org

Christopher Clark*, Kenton Lee*, Luke Zettlemoyer†*
{csquared,kentonl,lsz}@cs.washington.edu

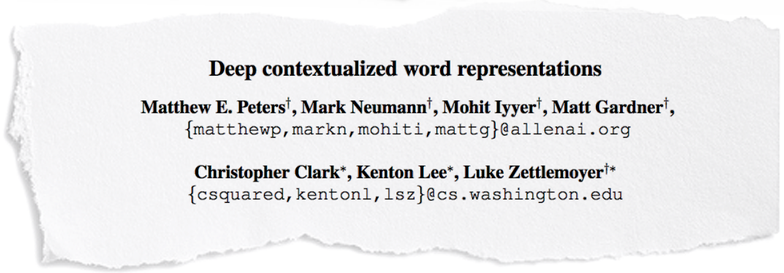
# Language model pretraining

- ULMFiT, ELMo: Predict the **next word** based on the previous words


Universal Language Model Fine-tuning for Text Classification

Jeremy Howard*
fast.ai
University of San Francisco
j@fast.ai

Sebastian Ruder*
Insight Centre, NUI Galway
Aylien Ltd., Dublin
sebastian@ruder.io


Deep contextualized word representations

Matthew E. Peters†, Mark Neumann†, Mohit Iyyer†, Matt Gardner†,
{matthewp,markn,mohiti,mattg}@allenai.org

Christopher Clark*, Kenton Lee*, Luke Zettlemoyer†*
{csquared,kentonl,lsz}@cs.washington.edu

- BERT: Predict a word given the **surrounding context**


BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Jacob Devlin Ming-Wei Chang Kenton Lee Kristina Toutanova
Google AI Language
{jacobdevlin,mingweichang,kentonl,kristout}@google.com

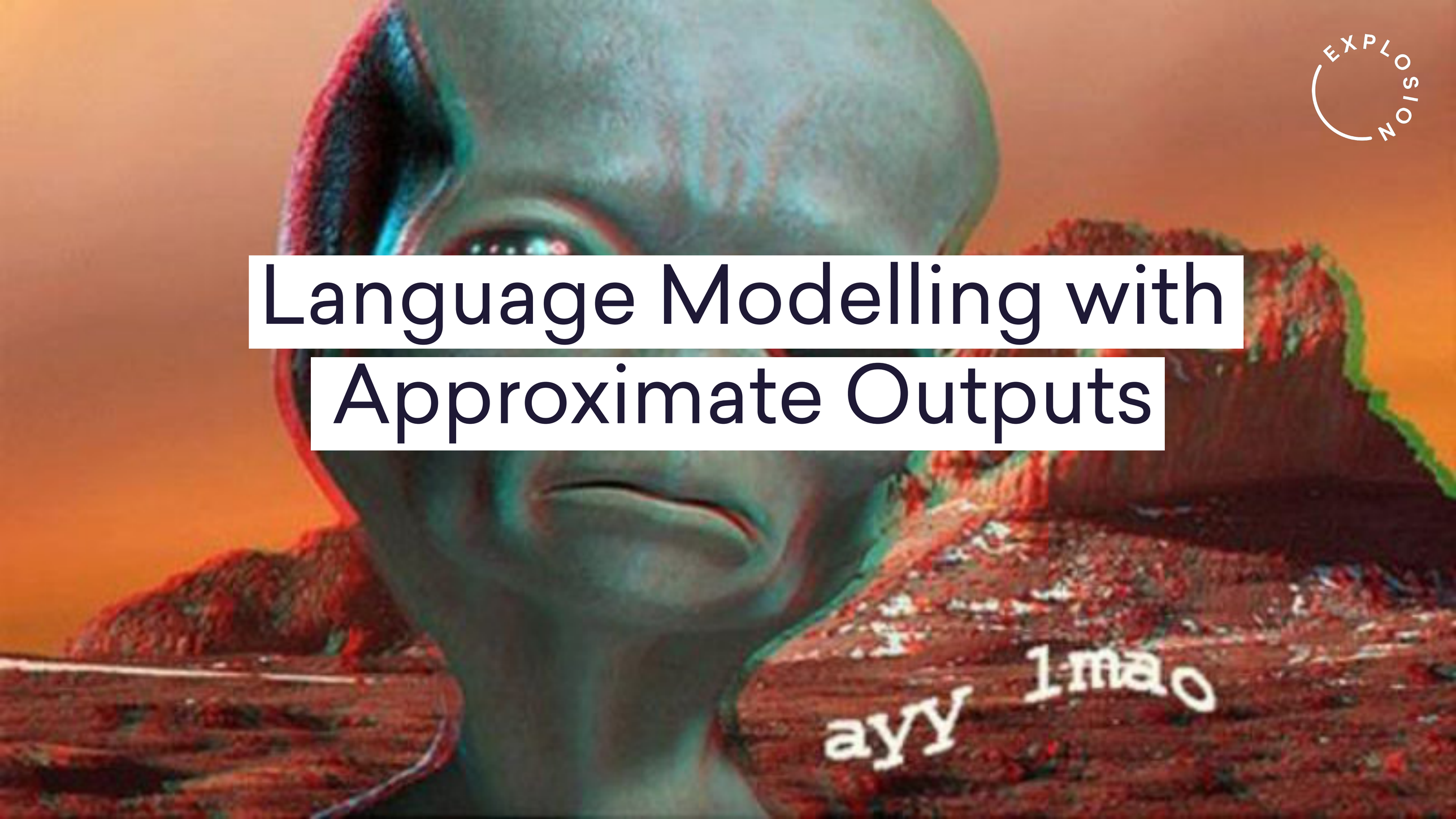
# Bringing language modelling into production

spaCy

Take what's proven to work in research, provide **fast, production-ready** implementations.

- Performance target: **10,000 words per second**
- Production models need to be **cheap to run** (and not require powerful GPUs)

# Language Modelling with Approximate Outputs



# 👽 Language Modelling with Approximate Outputs

- We train the CNN to predict the **vector of each word based on its context**
- Instead of predicting the *exact word*, we predict the **rough meaning** – much easier!
- Meaning representations learned with Word2Vec, GloVe or FastText

Kumar, Sachin, and Yulia Tsvetkov. "Von Mises-Fisher Loss for Training Sequence to Sequence Models with Continuous Outputs." arXiv preprint arXiv:1812.04616 (2019)

# Pretraining with spaCy

```
$ pip install spacy-nightly

$ spacy download en_vectors_web_lg

$ spacy pretrain ./reddit-100k.jsonl
  en_vectors_web_lg ./output_dir
```

# Pretraining with spaCy

```
$ pip install spacy-nightly

$ spacy download en_vectors_web_lg

$ spacy pretrain ./reddit-100k.jsonl
  en_vectors_web_lg ./output_dir
```

reddit-100k.jsonl


```
{"text": "Can I ask where you work now and what yo
{"text": "They may just pull out of the Seattle
{"text": "Its truly a great experience running 3
{"text": "Hue... hue... she has a front butt. :3"}
{"text": "My cynical view on this is that it will
{"text": "but gucci does this and no one says a
{"text": "La mul\u021bi ani! P\u0103i numai
{"text": "Mj faced good teams lol. And yes I can
{"text": "> almost valueless in modern (read that
{"text": "The only terms I couldn't figure out wer
{"text": "This is why I hate this subreddit
{"text": "rest in pepperoni's"}
```

# Pretraining with spaCy

```
$ pip install spacy-nightly

$ spacy download en_vectors_web_lg

$ spacy pretrain ./reddit-100k.jsonl
  en_vectors_web_lg ./output_dir

$ spacy train en ./model_out ./data/train
  ./data/dev --pipeline tagger,parser
  --init-tok2vec ./output_dir/model-best.t2v

✓ Saved best model to ./model_out/model-best
```

application.py

```
import spacy

nlp = spacy.load("./model_out/model-best")
doc = nlp("This is a sentence.")
for token in doc:
    print(token.text, token.pos_, token.dep_)
```

# Pretraining with spaCy

```
$ pip install spacy-nightly

$ spacy download en_vectors_web_lg

$ spacy pretrain ./reddit-100k.jsonl
  en_vectors_web_lg ./output_dir

$ spacy train en ./model_out ./data/train
  ./data/dev --pipeline tagger,parser
  --init-tok2vec ./output_dir/model-best.t2v

✓ Saved best model to ./model_out/model-best
```

| GloVe | LMAO | LAS |
|---|---|---|
| ❌ | ❌ | 79.1 |
| ✅ | ❌ | 81.0 |
| ❌ | ✅ | 81.0 |
| ✅ | ✅ | 82.4 |

Labelled attachment score (dependency parsing)
on Universal Dependencies data (English-EWT)

# Pretraining with spaCy

```
$ pip install spacy-nightly

$ spacy download en_vectors_web_lg

$ spacy pretrain ./reddit-100k.jsonl
  en_vectors_web_lg ./output_dir

$ spacy train en ./model_out ./data/train
  ./data/dev --pipeline tagger,parser
  --init-tok2vec ./output_dir/model-best.t2v

✓ Saved best model to ./model_out/model-best
```

| GloVe | LMAO | LAS |
|---|---|---|
| ❌ | ❌ | 79.1 |
| ✅ | ❌ | 81.0 |
| ❌ | ✅ | 81.0 |
| ✅ | ✅ | 82.4 |

3MB

| | |
|---|---|
| Stanford '17 | 82.3 |
| Stanford '18 | 83.9 |

Labelled attachment score (dependency parsing)
on Universal Dependencies data (English-EWT)

# 🚀 Move fast and train things

1. **Pre-train** models with general knowledge about the language using raw text.
2. **Annotate** a small amount of data specific to your application.
3. **Train** a model and try it in your application.
4. **Iterate** on your code *and* data.

# 🚀 Move fast and train things

1. **Pre-train** models with general knowledge about the language using raw text.
2. **Annotate** a small amount of data specific to your application.
3. **Train** a model and try it in your application.
4. **Iterate** on your code *and* data.

# Prodigy

https://prodi.gy

```
$ prodigy ner.teach product_ner
  en_core_web_sm /data.jsonl
  --label PRODUCT

$ prodigy db-out product_ner >
  annotations.jsonl
```


- **scriptable** annotation tool
- full **data privacy**: runs on your own hardware
- **active learning** for better example selection
- optimized for **efficiency** and fast **iteration**

# 🔄 Iterate on your code *and* your data

- Try out **more ideas** quickly. Most ideas *don't* work – but some succeed wildly.
- Figure out **what works** *before* trying to scale it up.
- Build entirely **custom solutions** so nobody can lock you in.

# Thanks!

💥 **Explosion AI**
explosion.ai

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