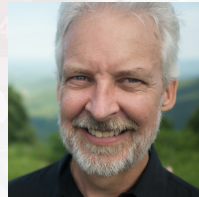


NLP SUMMIT

Presented by  John Snow LABS

Ray for NLP



Dean Wampler

Principal Software Engineer
Domino Data Lab (formerly Anyscale)

Domino Data Lab



Products ▾ Solutions ▾ Customers Resources ▾ Partners ▾ Company ▾

See Demo

Try Now

System-of-Record for Enterprise Data Science Teams



Accelerate Research

Get self-serve access to the latest tools and scalable compute. Reuse past work and iterate more efficiently.

[Learn More](#)



Centralize Infrastructure

Manage the availability of powerful data science resources in a secure and governed system-of-record.

[Learn More](#)



Deploy and Monitor Models

Expedite model consumption with apps, APIs, and more – and ensure their accuracy for key decisions.

[Learn More](#)



Unify Data Science Teams

Make data science teams more productive and collaborative, and manage their work more efficiently.

[Learn More](#)



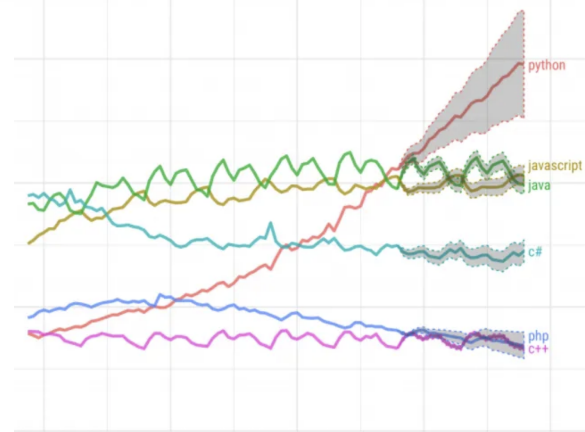
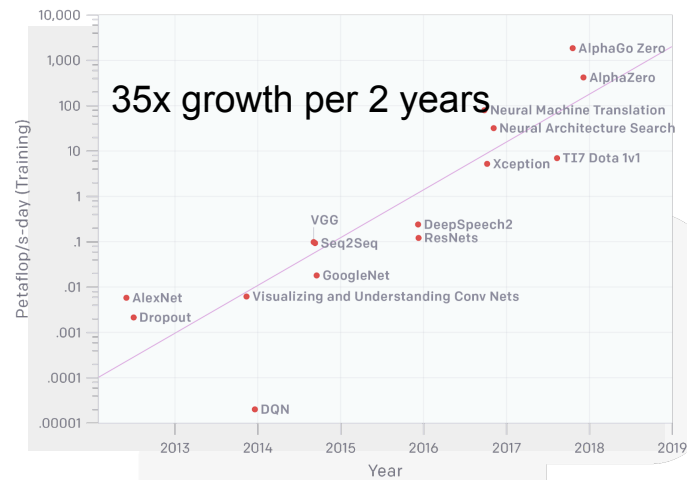
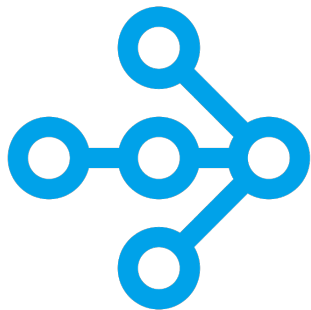
dominodatalab.com



@deanwampler

Why Ray

- Model sizes and compute requirements are growing rapidly.
- Python is the dominant data science programming language.
- ray.io

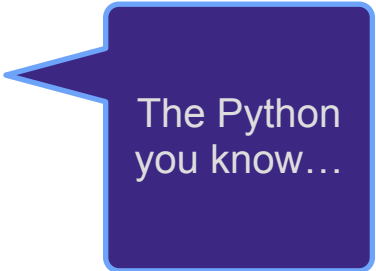


How to scale Python in N easy steps!

```
def make_array(...):  
    a = ... # Construct a NumPy array  
    return a
```

```
def add_arrays(a, b):  
    return np.add(a, b)
```

...



The Python
you know...

How to scale Python in N easy steps!

```
@ray.remote  
def make_array(...):  
    a = ... # Construct a NumPy array  
    return a
```

```
@ray.remote  
def add_arrays(a, b):  
    return np.add(a, b)  
...
```

Turn a
function into
a **task**.

for completeness, start with:

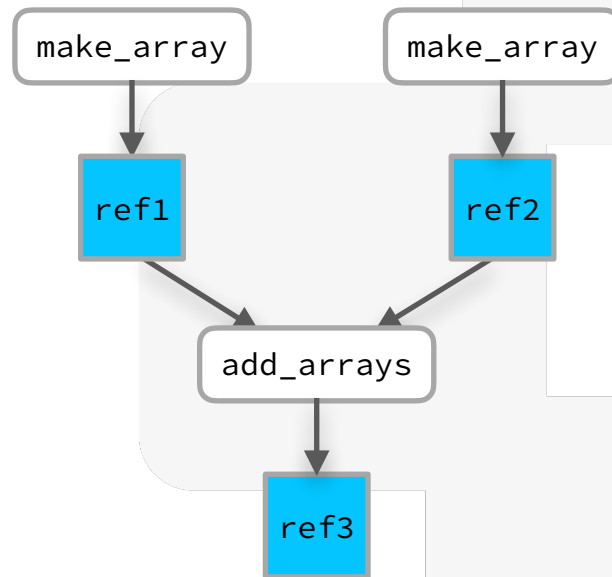
```
import ray  
import numpy as np
```

```
ray.init(...)
```

How to scale Python in N easy steps!

```
...  
ref1 = make_array.remote(...)  
ref2 = make_array.remote(...)  
ref3 = add_arrays.remote(ref1, ref2)
```

Start a task
with `remote`.

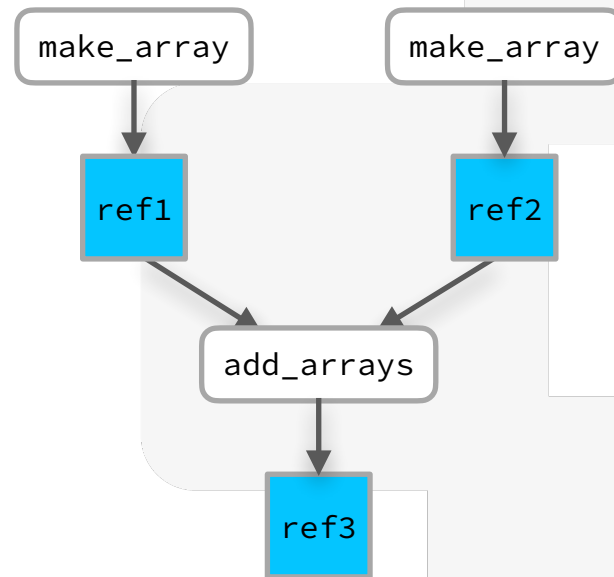


How to scale Python in N easy steps!

...

```
ref1 = make_array.remote(...)  
ref2 = make_array.remote(...)  
ref3 = add_arrays.remote(ref1, ref2)  
array = ray.get(ref3)
```

Fetch the
computed
value



How to scale Python in N easy steps!

...

```
ref1 = make_array.remote(...)
ref2 = make_array.remote(...)
ref3 = add_arrays.remote(ref1, ref2)
array = ray.get(ref3)
```

Ray handles
cluster
scheduling,
async
computing

No need to
call ray.get()
for these
first!

How to scale Python in N easy steps!

...

```
ref1 = make_array.remote(...)  
ref2 = make_array.remote(...)  
ref3 = add_arrays.remote(ref1, ref2)  
array = ray.get(ref3)
```

What about
Distributed
state??

How to scale Python in N easy steps!

...

```
ref1 = make_array.remote(...)
ref2 = make_array.remote(...)
ref3 = add_arrays.remote(ref1, ref2)
array = ray.get(ref3)
```

The Python
classes you
love

```
class Counter(object):
    def __init__(self):
        self.value = 0
    def increment(self):
        self.value += 1
        return self.value
```

How to scale Python in N easy steps!

```
...  
ref1 = make_array.remote(...)  
ref2 = make_array.remote(...)  
ref3 = add_arrays.remote(ref1, ref2)  
array = ray.get(ref3)
```

`@ray.remote`

```
class Counter(object):  
    def __init__(self):  
        self.value = 0  
    def increment(self):  
        self.value += 1  
        return self.value  
    def get_count(self):  
        return self.value
```

From class to
“actor”

Must add a
getter
method

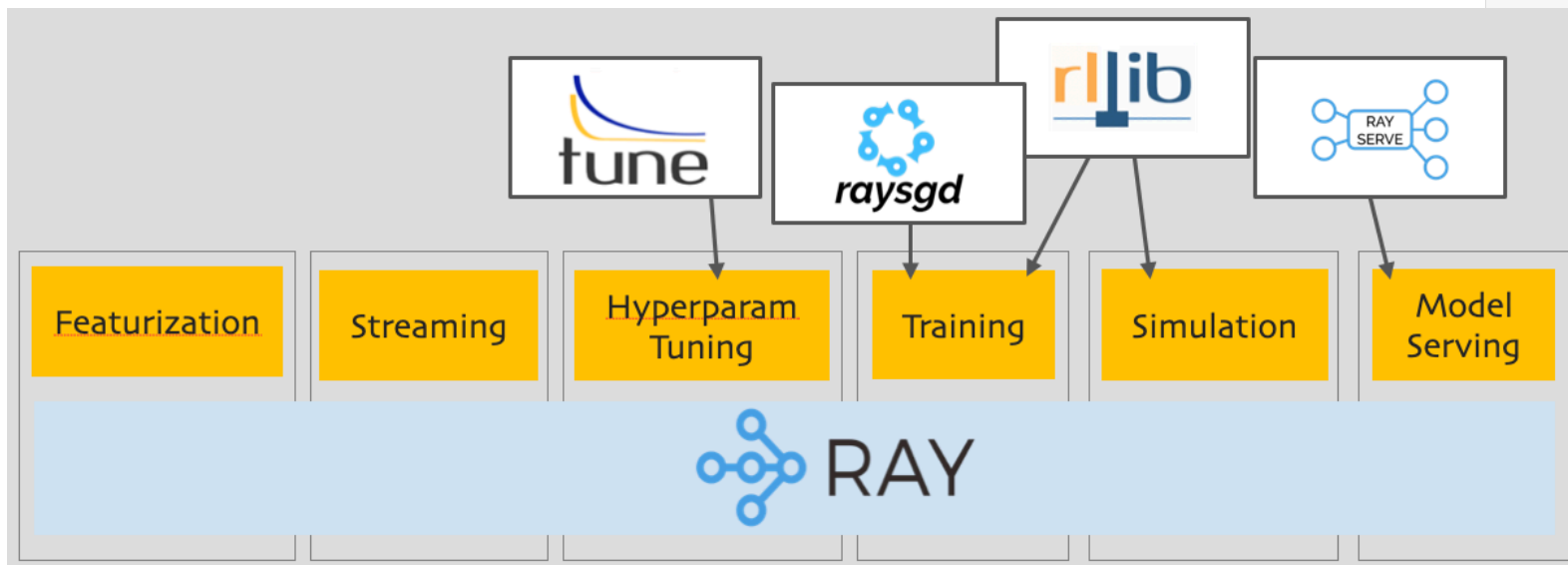
How to scale Python in N easy steps!

```
...  
ref1 = make_array.remote(...)  
ref2 = make_array.remote(...)  
ref3 = add_arrays.remote(ref1, ref2)  
array = ray.get(ref3)
```

```
...  
c = Counter.remote()  
ref4 = c.increment.remote()  
ref5 = c.increment.remote()  
ray.get([ref4,ref5]) # [1, 2]
```

Same idioms

But you may never use the Ray API...



ray.io

But you may never use the Ray API...



huggingface.co

Hugging Face Transformers

Since NLP model training is \$\$\$\$\$, it's easier to use **transfer**

learning:

- Start with a pre-trained model
- Add a few more layers
- Train for a few epochs for a particular application
- Profit?



Transformers

github.com/huggingface/transformers

Hugging Face Transformers

Well, hyper-parameter tuning is also expensive and it can be tricky.

- Avoiding local minima: arxiv.org/abs/1811.01088
- High variance in results common: github.com/pytorch/fairseq/blob/master/examples/roberta/wsc

credit: Thomas Wolf, *Transfer Learning in NLP: Concepts, Tools & Trends* (Ray Summit 2020)



Transformers

github.com/huggingface/transformers

Hugging Face Transformers

Using [Ray Tune](#) You can get 1.5% better results using *Bayesian Optimization*, 5% better using *Population-Based Training* for the same compute resources.

- See this [blog post](#) ([Ray blog](#) on Medium)

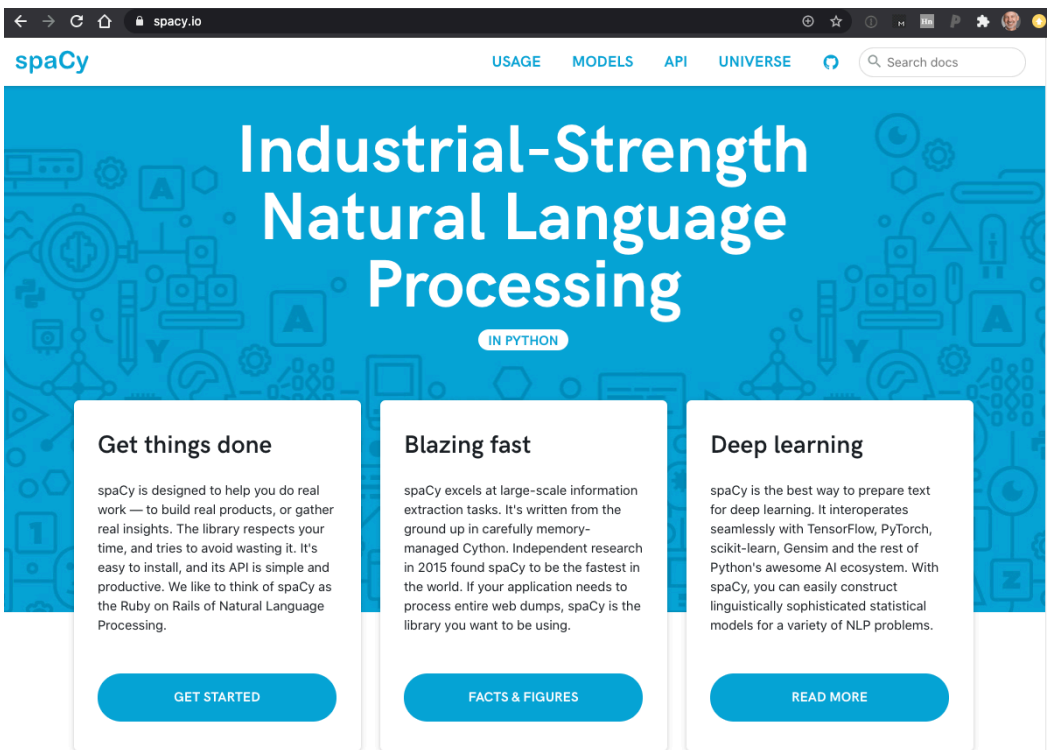
Hyper-parameters:

- learning rate
- weight decay
- # of epochs
- per-GPU batch size



tune.io

But you may never use the Ray API...



The screenshot shows the spaCy website homepage. The browser address bar displays 'spacy.io'. The navigation menu includes 'USAGE', 'MODELS', 'API', and 'UNIVERSE', along with a search bar labeled 'Search docs'. The main heading is 'Industrial-Strength Natural Language Processing' with a sub-heading 'IN PYTHON'. Below this are three columns of text, each with a 'GET STARTED', 'FACTS & FIGURES', or 'READ MORE' button.

spaCy

USAGE MODELS API UNIVERSE Search docs

Industrial-Strength Natural Language Processing

IN PYTHON

Get things done

spaCy is designed to help you do real work — to build real products, or gather real insights. The library respects your time, and tries to avoid wasting it. It's easy to install, and its API is simple and productive. We like to think of spaCy as the Ruby on Rails of Natural Language Processing.

GET STARTED

Blazing fast

spaCy excels at large-scale information extraction tasks. It's written from the ground up in carefully memory-managed Cython. Independent research in 2015 found spaCy to be the fastest in the world. If your application needs to process entire web dumps, spaCy is the library you want to be using.

FACTS & FIGURES

Deep learning

spaCy is the best way to prepare text for deep learning. It interoperates seamlessly with TensorFlow, PyTorch, scikit-learn, Gensim and the rest of Python's awesome AI ecosystem. With spaCy, you can easily construct linguistically sophisticated statistical models for a variety of NLP problems.

READ MORE

spacy.io

spaCy v3

spaCy v3 release will introduce a new integration with Ray, which will bring effortless parallel and distributed training to spaCy.

- github.com/explosion/spacy-ray
- Matthew Honnibal, [Why spaCy Is Going with Ray](#) (Ray Summit 2020)



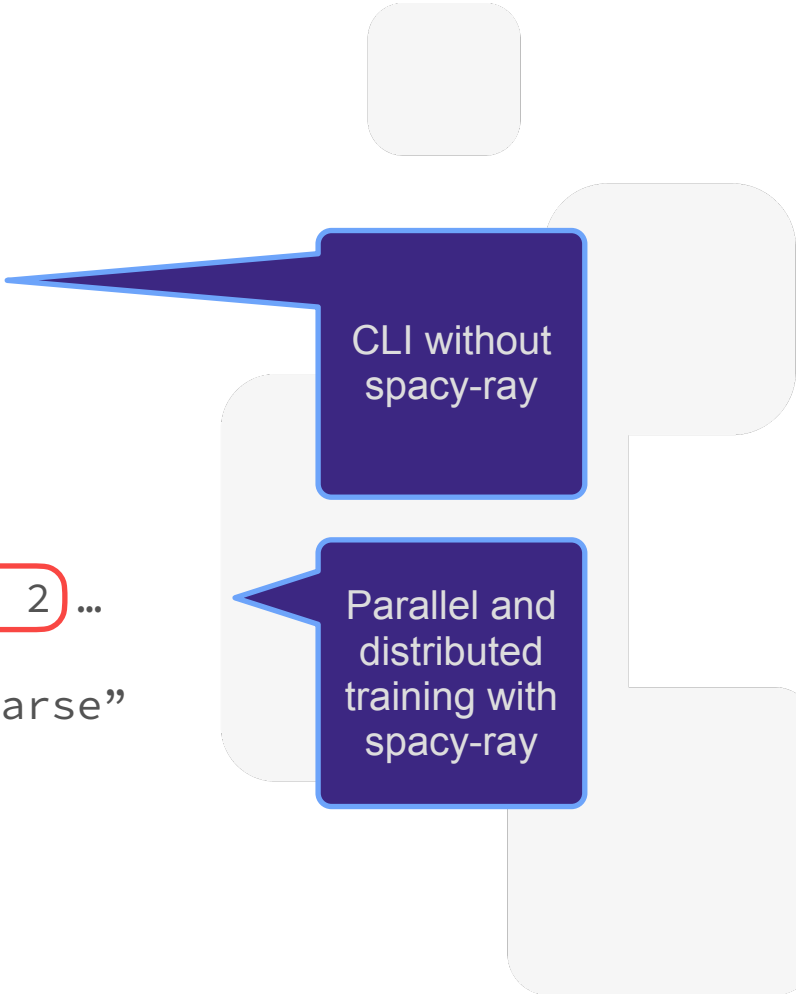
spaCy v3

```
$ python -m spacy train ...
```

```
$ pip install spacy-ray
```

```
$ python -m spacy ray train --n-workers 2 ...
```

```
# “spacy ray pretrain” and “spacy ray parse”  
# are planned.
```



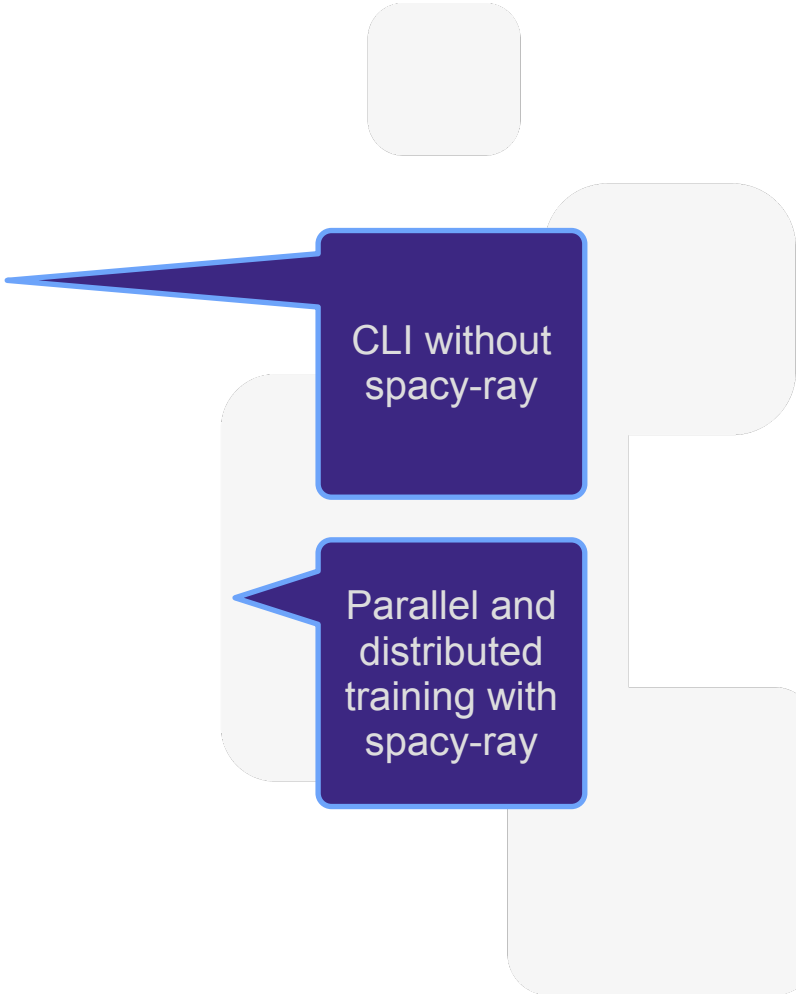
CLI without
spacy-ray

Parallel and
distributed
training with
spacy-ray

spaCy v3

spaCy v3 includes changes to the data model and some pipeline improvements. The Ray support is not a lot of code:

- Shard parameters into distributed state with Ray actors.
- Train on local shard.
- Asynchronously receive updates from other actors.
- Merge updates.
- Repeat...



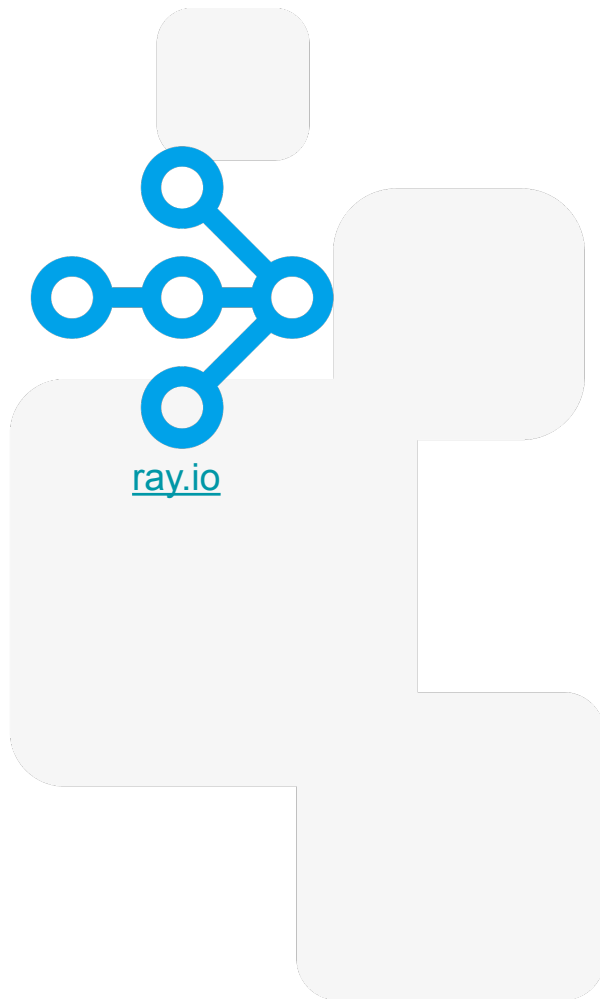
CLI without
spacy-ray

Parallel and
distributed
training with
spacy-ray

Your uses of Ray for NLP?

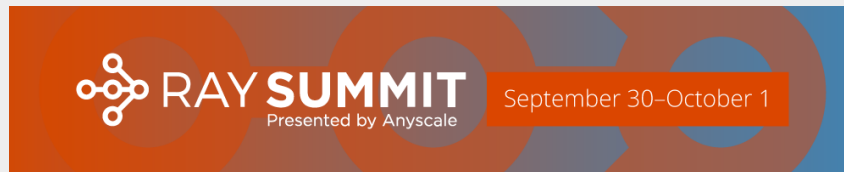
Ray's flexible *task* model can be used for course- and fine-grained computation. The *actor* model makes “sharded”, distributed state intuitive to manage. So, use it for:

- Tokenization and other data prep
- Distributed training: Ray Tune and [Ray SGD](#) for easier distributed TensorFlow and PyTorch
- Simple, scalable model serving with [Ray Serve](#)



Ray in NLP

The creators of Hugging Face and spaCy and the how they use Ray. See the [Anyscale blog](#) and [YouTube channel](#).



raysummit.org

Domino Data Lab



Products ▾ Solutions ▾ Customers Resources ▾ Partners ▾ Company ▾

See Demo

Try Now

System-of-Record for Enterprise Data Science Teams



Accelerate Research

Get self-serve access to the latest tools and scalable compute. Reuse past work and iterate more efficiently.

[Learn More](#)



Centralize Infrastructure

Manage the availability of powerful data science resources in a secure and governed system-of-record.

[Learn More](#)



Deploy and Monitor Models

Expedite model consumption with apps, APIs, and more – and ensure their accuracy for key decisions.

[Learn More](#)



Unify Data Science Teams

Make data science teams more productive and collaborative, and manage their work more efficiently.

[Learn More](#)



dominodatalab.com



@deanwampler

Thanks for listening!

- ray.io
- raysummit.org
- dean@dominodatalab.com
- [@deanwampler](https://twitter.com/deanwampler)

**NLP
SUMMIT**

Presented by  **John Snow LABS**

Title

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.