Taming the Deep Learning Workflow

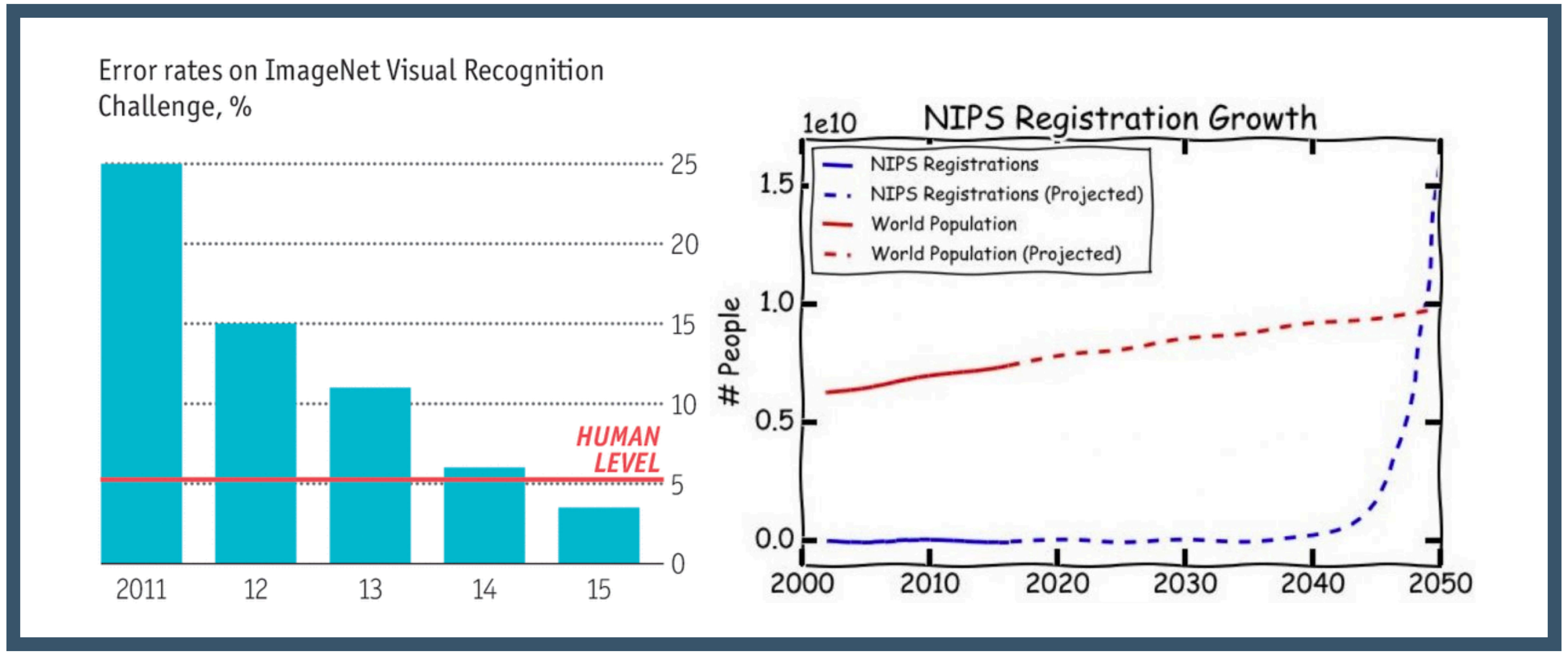
Evan Sparks

March 18, 2019

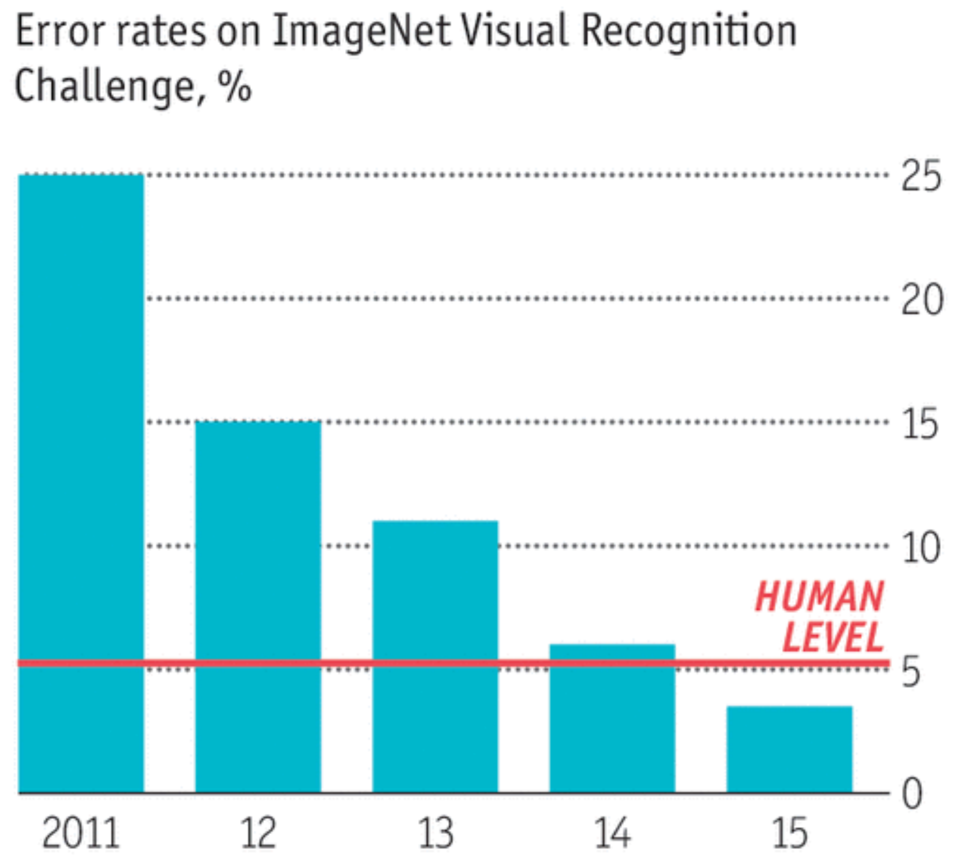
Determined AI

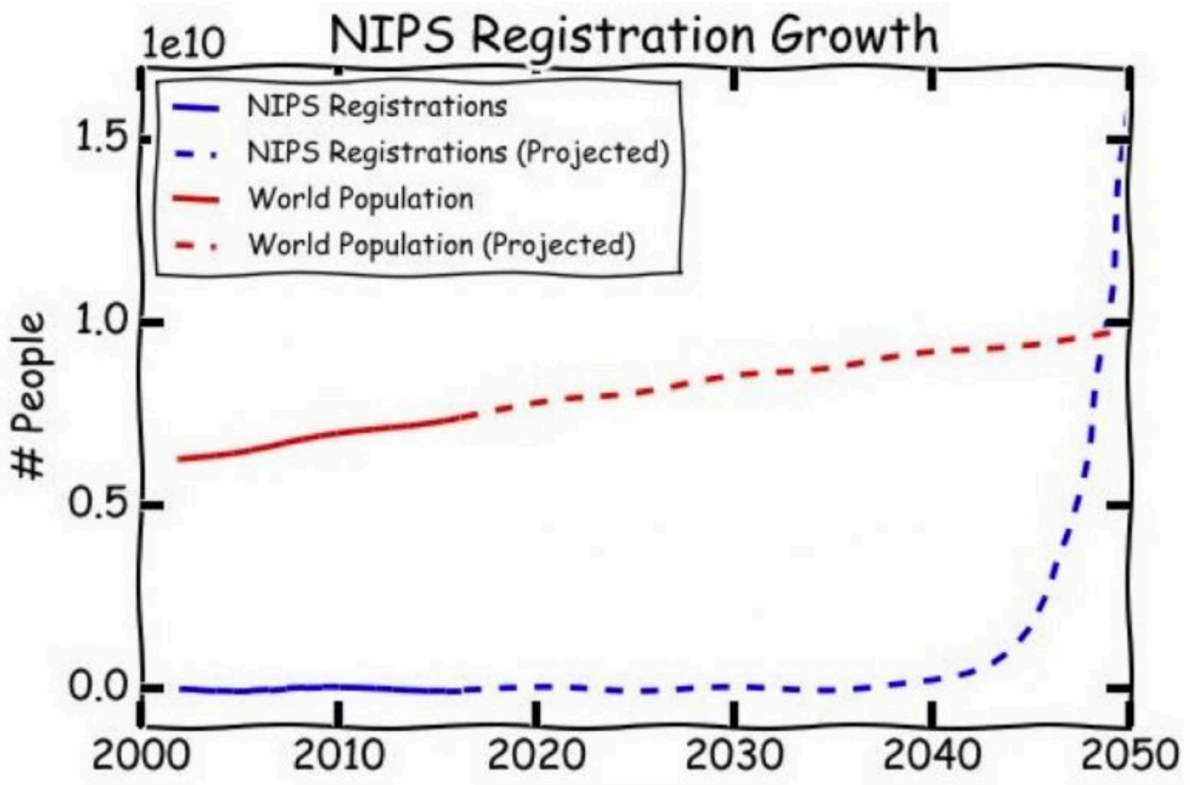
The Golden Age of AI

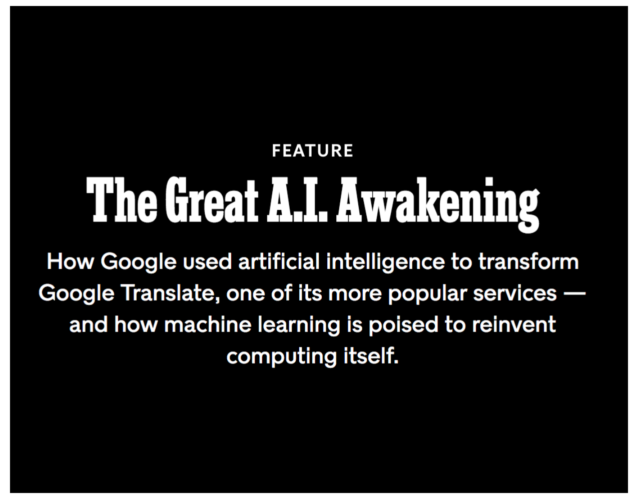
The Golden Age of AI



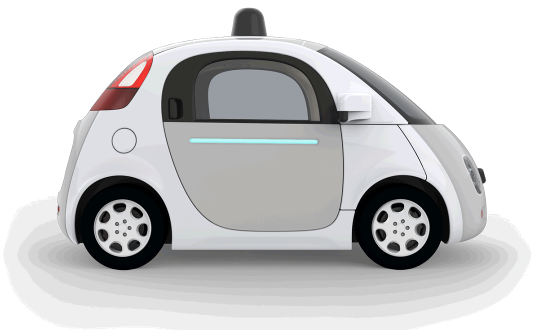
The Golden Age of AI



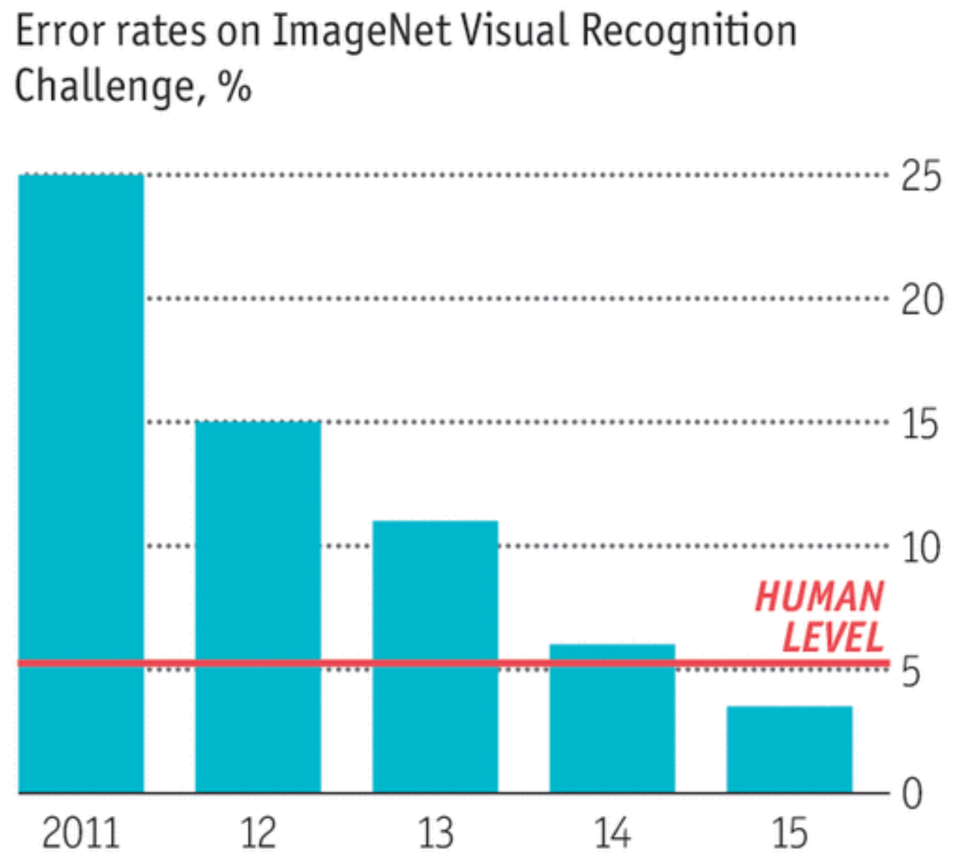


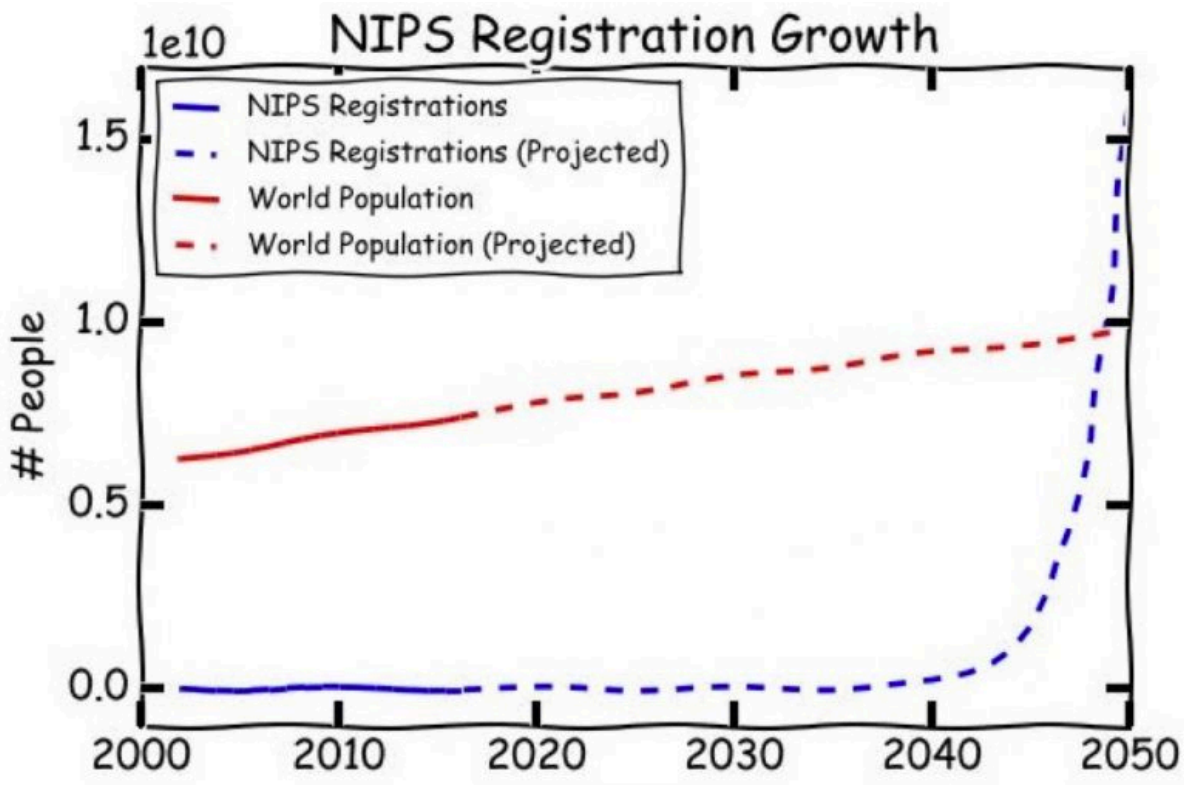

FEATURE
The Great A.I. Awakening
How Google used artificial intelligence to transform Google Translate, one of its more popular services — and how machine learning is poised to reinvent computing itself.

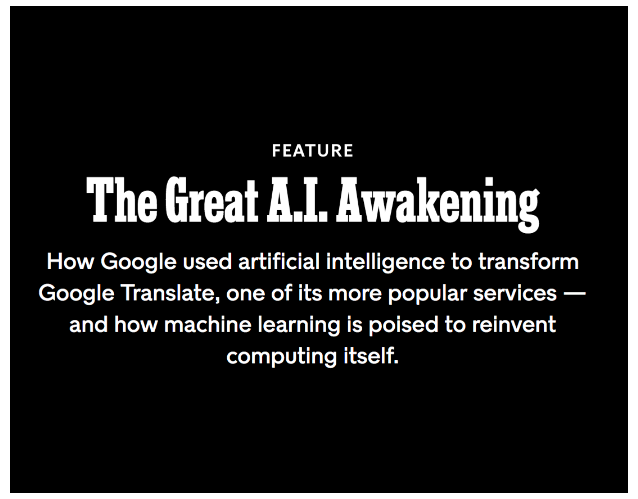

Tech Giants Are Paying Huge Salaries for Scarce A.I. Talent
Nearly all big tech companies have an artificial intelligence project, and they are willing to pay experts millions of dollars to help get it done.



The Golden Age of AI





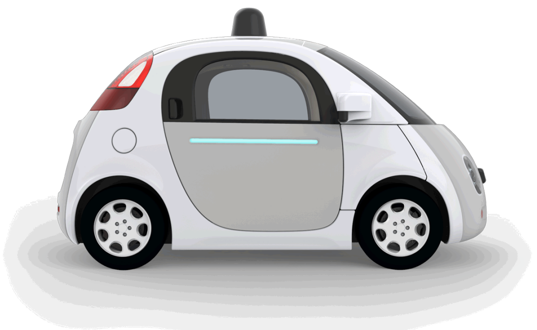

FEATURE

The Great A.I. Awakening

How Google used artificial intelligence to transform Google Translate, one of its more popular services — and how machine learning is poised to reinvent computing itself.


Tech Giants Are Paying Huge Salaries for Scarce A.I. Talent

Nearly all big tech companies have an artificial intelligence project, and they are willing to pay experts millions of dollars to help get it done.



Technology

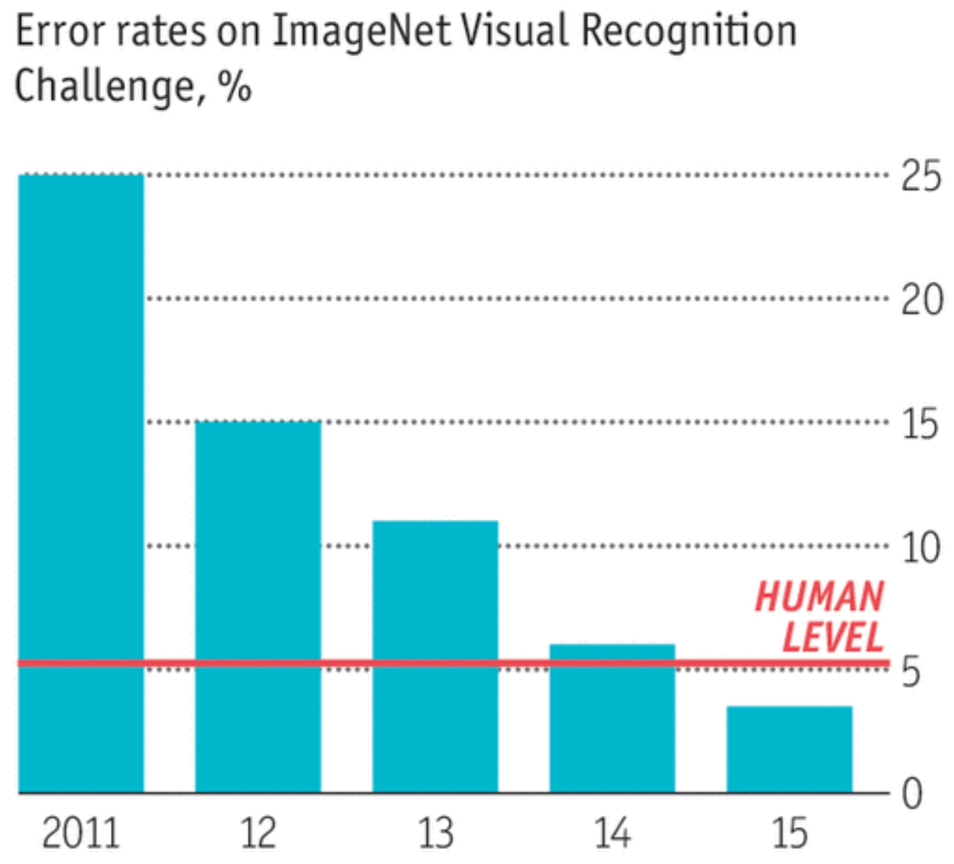
Coming This Fall to Carnegie Mellon: America's First AI Degree

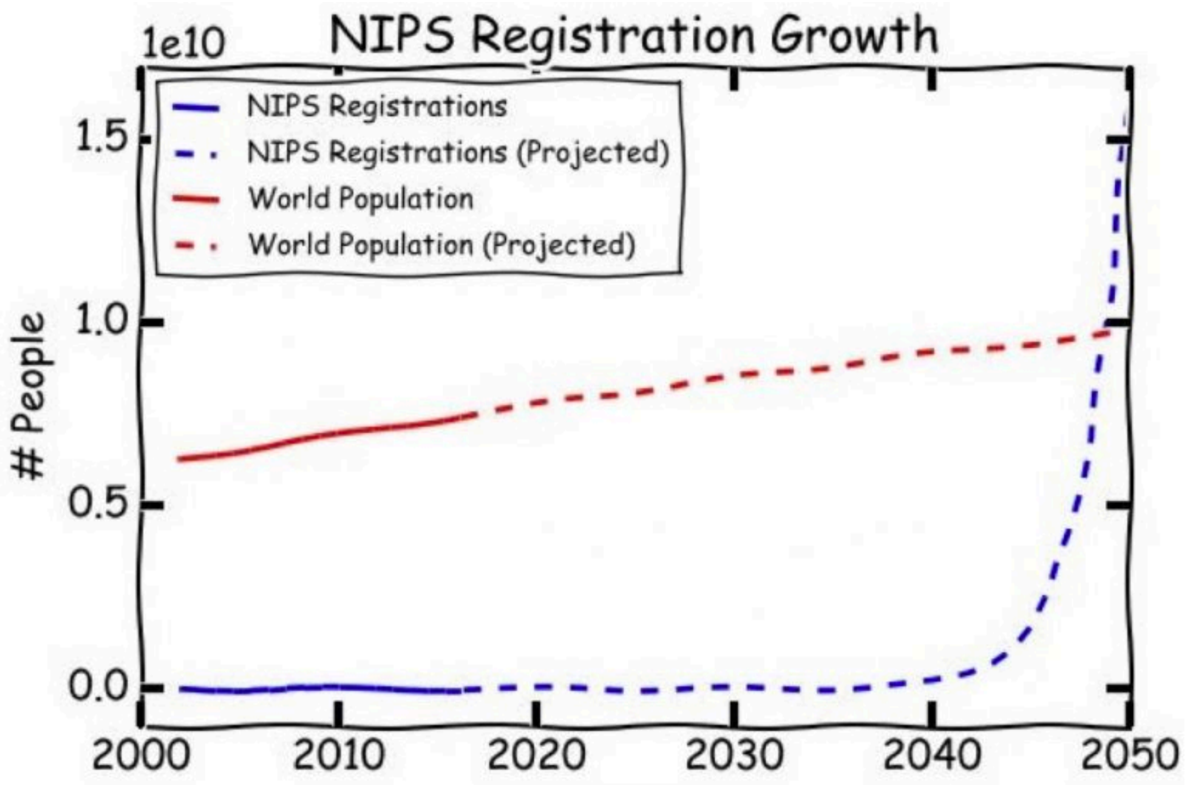
TECHNOLOGY

Stanford's Top Major Is Now Computer Science

Keras4Kindergartners

The Golden Age of AI





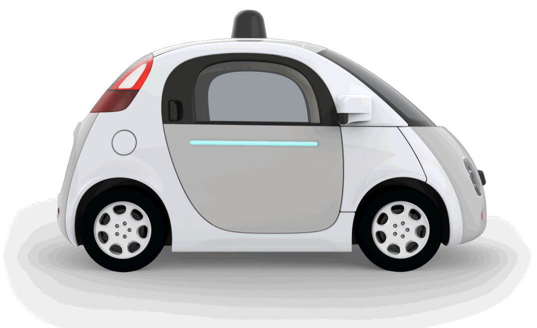
FEATURE

The Great A.I. Awakening

How Google used artificial intelligence to transform Google Translate, one of its more popular services — and how machine learning is poised to reinvent computing itself.

Tech Giants Are Paying Huge Salaries for Scarce A.I. Talent

Nearly all big tech companies have an artificial intelligence project, and they are willing to pay experts millions of dollars to help get it done.



Technology

Coming This Fall to Carnegie Mellon: America's First AI Degree

TECHNOLOGY

Stanford's Top Major Is Now Computer Science

Keras4Kindergartners

AI ready for widespread adoption?

The Dark Age of AI Infrastructure




The Dark Age of AI Infrastructure

Forcing users to wait for **days** to recover from faults.




The Dark Age of AI Infrastructure

Forcing users to wait for **days** to recover from faults.

Reproducing existing models is **death by a thousand cuts:** data ordering, software versions, hyperaparmeters, random seeds, model weights.

The Dark Age of AI Infrastructure

Forcing users to wait for **days** to recover from faults.

Reproducing existing models is **death by a thousand cuts:** data ordering, software versions, hyperaparmeters, random seeds, model weights.

Hand-implemented, **impossibly slow** methods to find good models.

The Dark Age of AI Infrastructure

Forcing users to wait for **days** to recover from faults.

Reproducing existing models is **death by a thousand cuts:** data ordering, software versions, hyperaparmeters, random seeds, model weights.

Hand-implemented, **impossibly slow** methods to find good models.

Trapping our users in systems designed to house **one user** with **rigid infrastructure.**

Deep Learning Today (For Everyone Else)

Existing Tools (e.g., TensorFlow):

- Mostly focused on 1 researcher training 1 model on 1 GPU

Limited Support For:

- Teams of researchers, clusters of GPUs, many models
- Deployment, ops, and collaboration

We need **holistic** and **specialized** AI software infrastructure.

The Dark Age of AI Infrastructure

Forcing users to wait for **days** to recover from faults.

Reproducing existing models is **death by a thousand cuts:** data ordering, software versions, hyperaparmeters, random seeds, model weights.

Hand-implemented, **impossibly slow** methods to find good models.

Trapping our users in systems designed to house **one user** with **rigid infrastructure.**

The Dark Age of AI Infrastructure

Forcing users to wait for **days** to recover from faults.

Reproducing existing models is **death by a thousand cuts:** data ordering, software versions, hyperaparmeters, random seeds, model weights.

Hand-implemented, **impossibly slow** methods to find good models.

Trapping our users in systems designed to house **one user** with **rigid infrastructure.**

Dave’s got a problem.

Dave's got a problem.

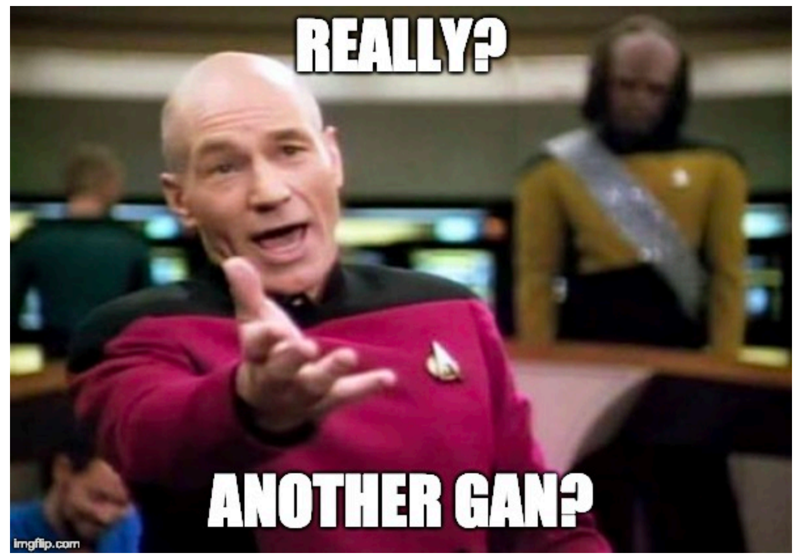
- Dave's a super smart DL engineer.

Dave's got a problem.

- Dave's a super smart DL engineer.
- He's got a brilliant model for style transfer that automatically makes every picture a dank meme.

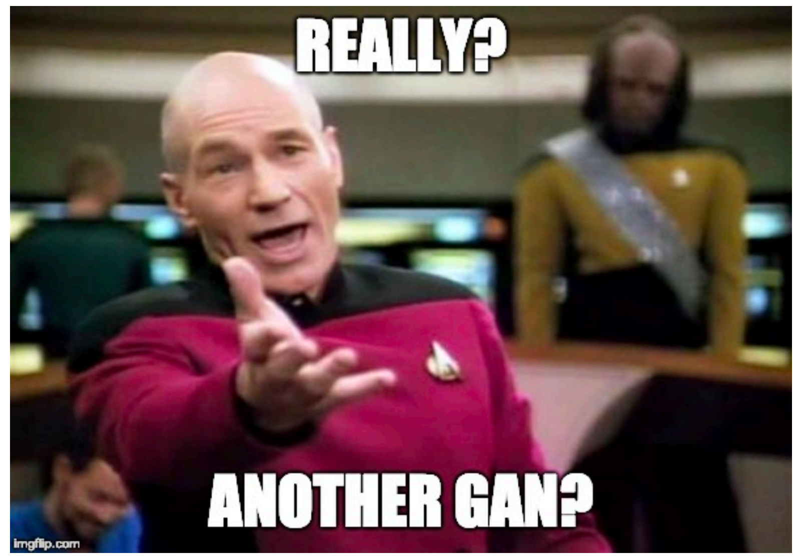
Dave's got a problem.

- Dave's a super smart DL engineer.
- He's got a brilliant model for style transfer that automatically makes every picture a dank meme.



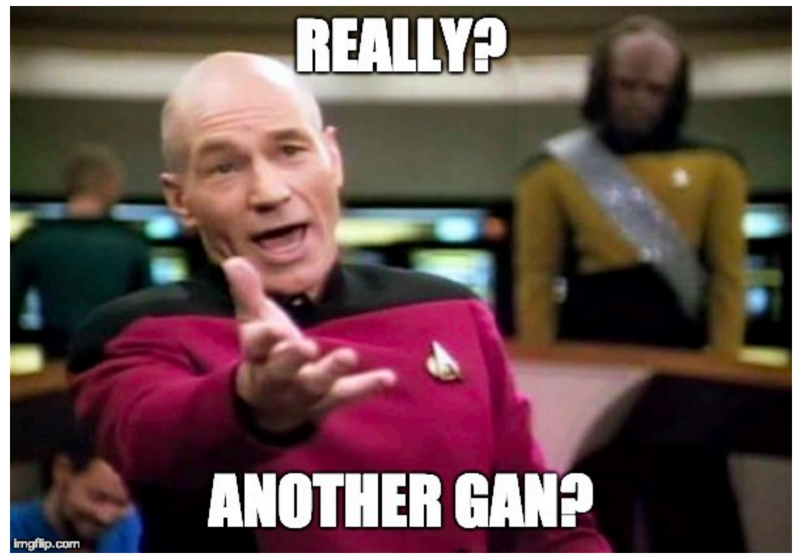
Dave's got a problem.

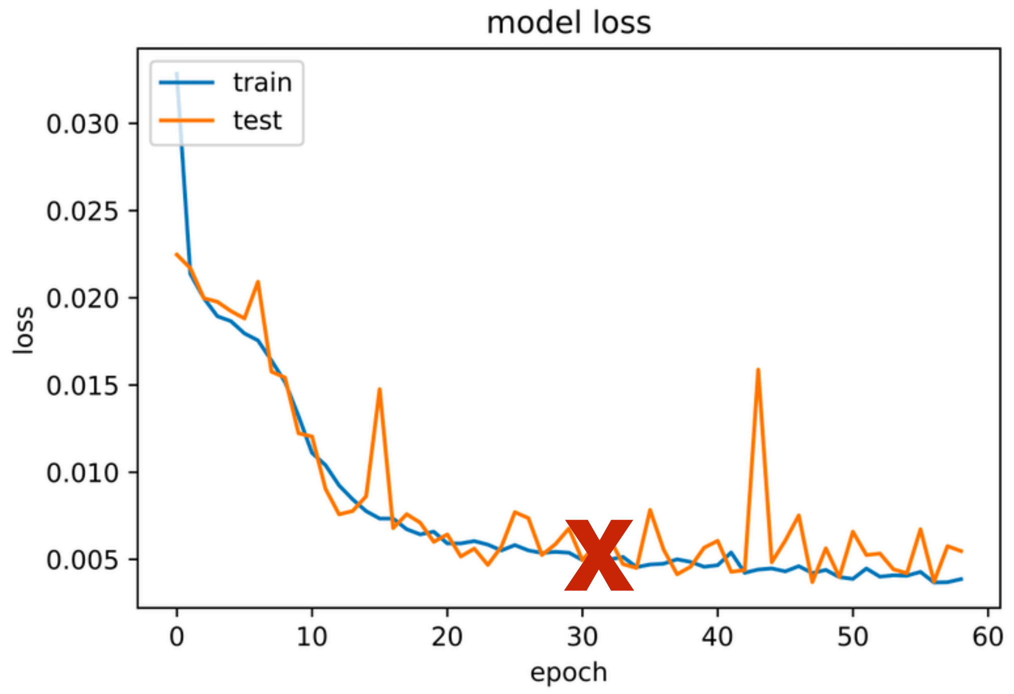
- Dave's a super smart DL engineer.
- He's got a brilliant model for style transfer that automatically makes every picture a dank meme.
- It takes two days for his model to converge on a couple of DGX-1s.



Dave's got a problem.

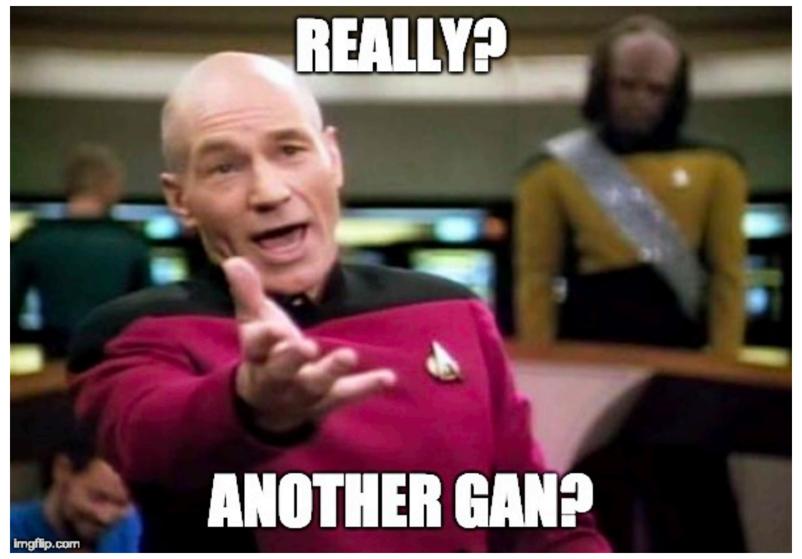
- Dave's a super smart DL engineer.
- He's got a brilliant model for style transfer that automatically makes every picture a dank meme.
- It takes two days for his model to converge on a couple of DGX-1s.
- Every time his model crashes he loses (on average) a day of work and 400 GPU-hours of compute time.

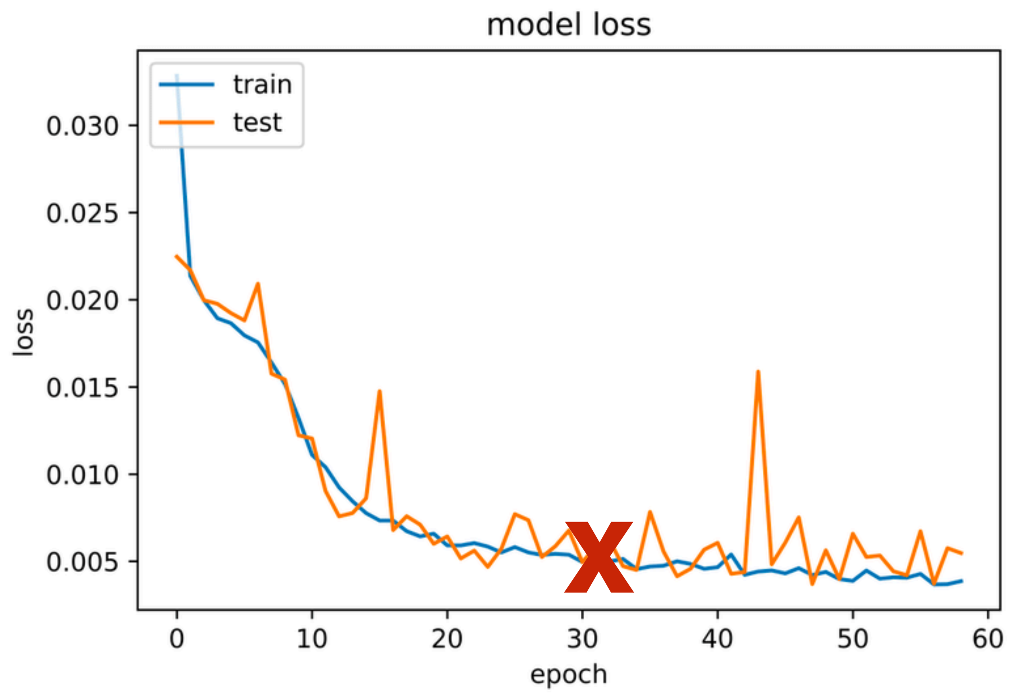




Dave's got a problem.

- Dave's a super smart DL engineer.
- He's got a brilliant model for style transfer that automatically makes every picture a dank meme.
- It takes two days for his model to converge on a couple of DGX-1s.
- Every time his model crashes he loses (on average) a day of work and 400 GPU-hours of compute time.
- This makes Dave sad.





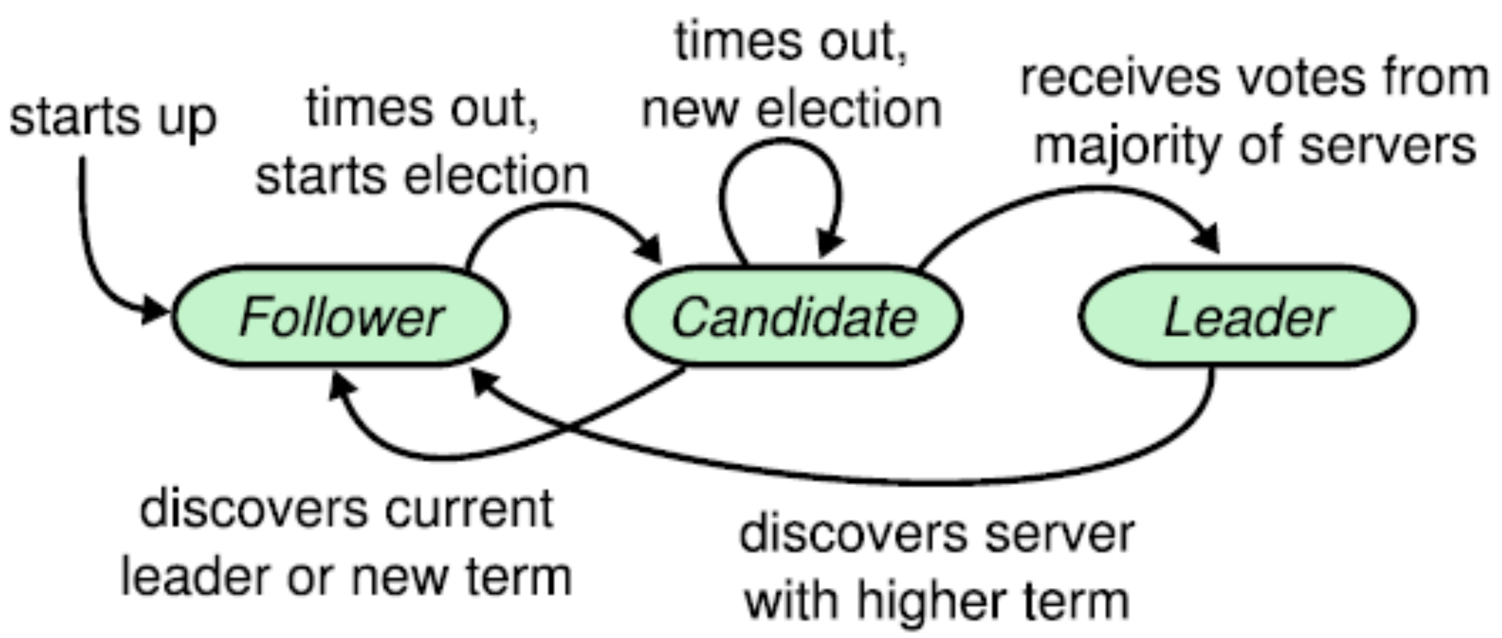
Dave's got a solution!

Dave's got a solution!

- What Dave wants is a way to make sure he doesn't lose work.

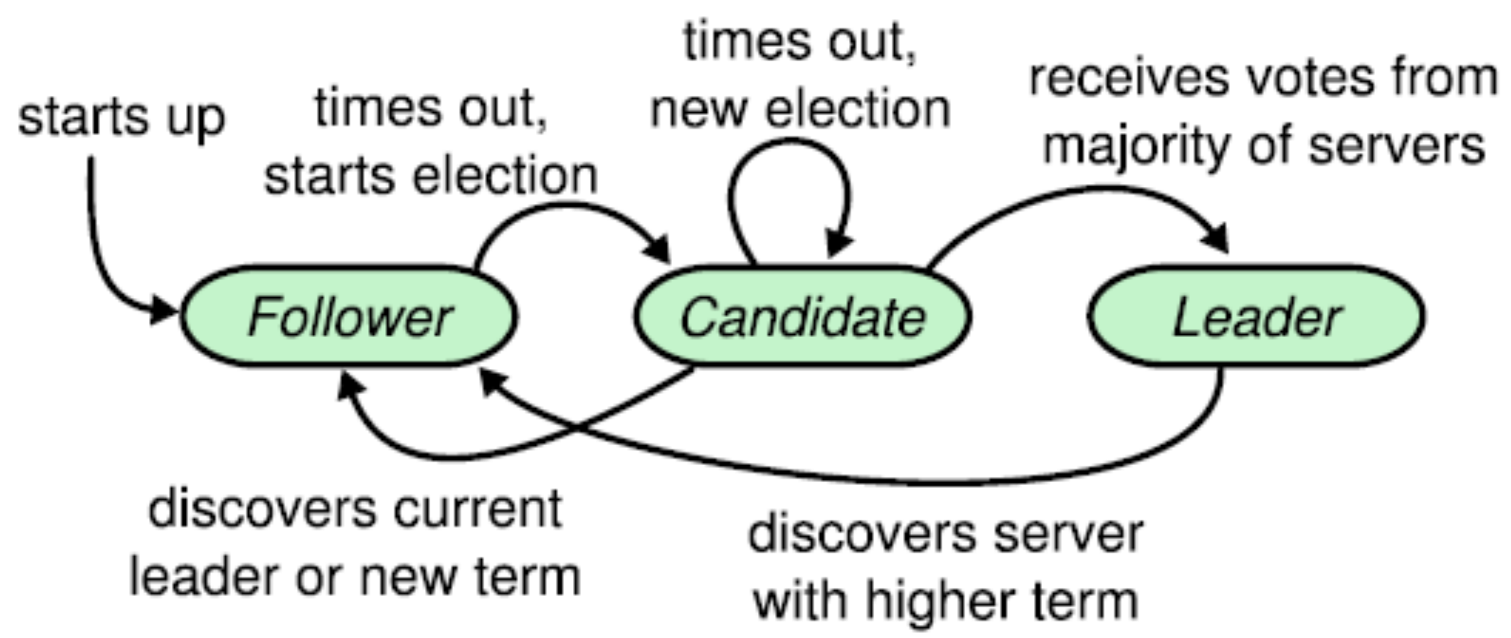
Dave's got a solution!

- What Dave wants is a way to make sure he doesn't lose work.
- In general, this is a "hard problem."



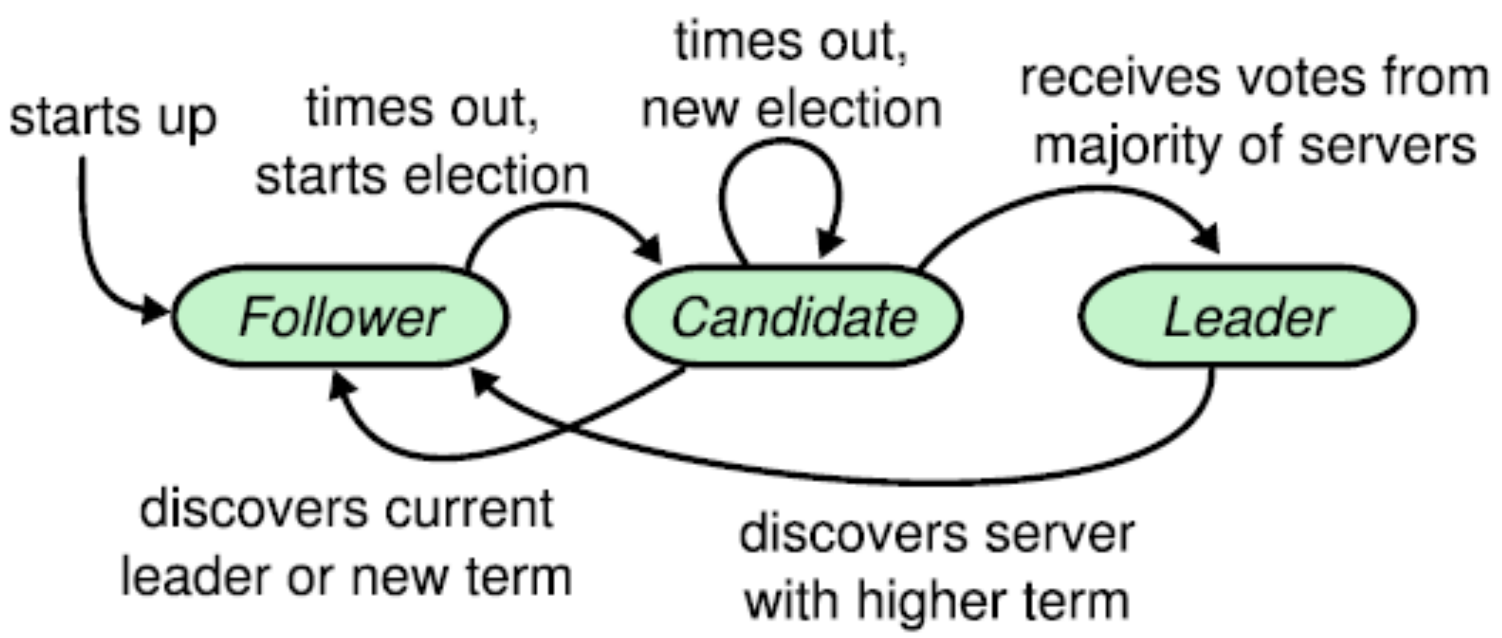
Dave's got a solution!

- What Dave wants is a way to make sure he doesn't lose work.
- In general, this is a "hard problem."
- In Deep Learning - this isn't so bad. Enter `tf.saved_model.simple_save`.



Dave's got a solution!

- What Dave wants is a way to make sure he doesn't lose work.
- In general, this is a "hard problem."
- In Deep Learning - this isn't so bad. Enter `tf.saved_model.simple_save`.
- So, Dave instruments his code, and the next time it crashes he loads his model using `tf.saved_model.loader.load` and keeps on training.



Only, he doesn’t.

Only, he doesn't.

- TensorFlow only saves:
  - Weights, optimizer state.

Only, he doesn't.

- TensorFlow only saves:
  - Weights, optimizer state.
- Dave also needs
  - TF Version, input read position, random seeds, model definition.

Only, he doesn't.

- TensorFlow only saves:
  - Weights, optimizer state.
- Dave also needs
  - TF Version, input read position, random seeds, model definition.
- Eventually, **Dave writes a pile of code to save all this stuff.**

And Dave's life <u>still sucks</u>

And Dave's life <u>still sucks</u>

- Learns the hard way that checkpoints are really big, and runs out of disk space.

And Dave's life <u>still sucks</u>

- Learns the hard way that checkpoints are really big, and runs out of disk space.
- Teaches himself PagerDuty so that he can find out when his models crash and ssh back into work to kick the models off.

And Dave's life <u>still sucks</u>

- Learns the hard way that checkpoints are really big, and runs out of disk space.
- Teaches himself PagerDuty so that he can find out when his models crash and ssh back into work to kick the models off.
- Loses his place in the queue.

And Dave's life still sucks

- Learns the hard way that checkpoints are really big, and runs out of disk space.
- Teaches himself PagerDuty so that he can find out when his models crash and ssh back into work to kick the models off.
- Loses his place in the queue.
- **Dave writes a pile of cron jobs to make sure his work is being done.**

What if Dave had holistic but specialized AI infrastructure?

What if Dave had holistic but specialized AI infrastructure?

- Checkpointing would be taken care of (the right way) out of the box.
- The **infrastructure** would monitor and retry failed jobs from latest checkpoint automatically.
- The **infrastructure** would manage its own checkpoint storage according to sane rules (keep models with the best n validation errors).
- The **infrastructure** could leverage checkpoints in other, surprising ways: to enable reproducibility, as a unit of scheduling/job migration, and to enable distributed training.
- All of this would be **transparent** to Dave.

The Dark Age of AI Infrastructure

Forcing users to wait for **days** to recover from faults.

Reproducing existing models is **death by a thousand cuts:** data ordering, software versions, hyperaparmeters, random seeds, model weights.

Hand-implemented, **impossibly slow** methods to find good models.

Trapping our users in systems designed to house **one user** with **rigid infrastructure.**

The Dark Age of AI Infrastructure

Forcing users to wait for **days** to recover from faults.

Reproducing existing models is **death by a thousand cuts:** data ordering, software versions, hyperaparmeters, random seeds, model weights.

Hand-implemented, **impossibly slow** methods to find good models.

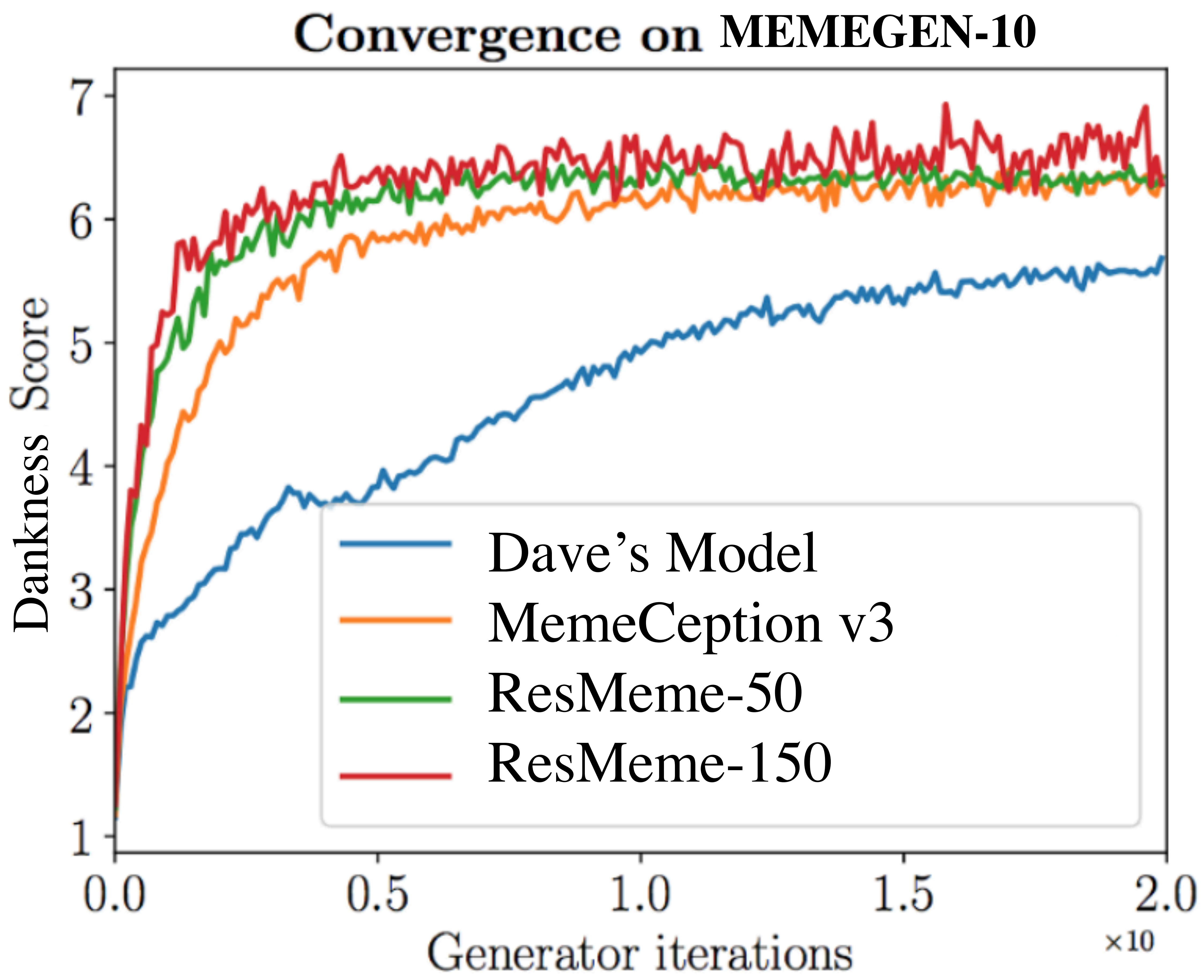
Trapping our users in systems designed to house **one user** with **rigid infrastructure.**

Dave trains his model

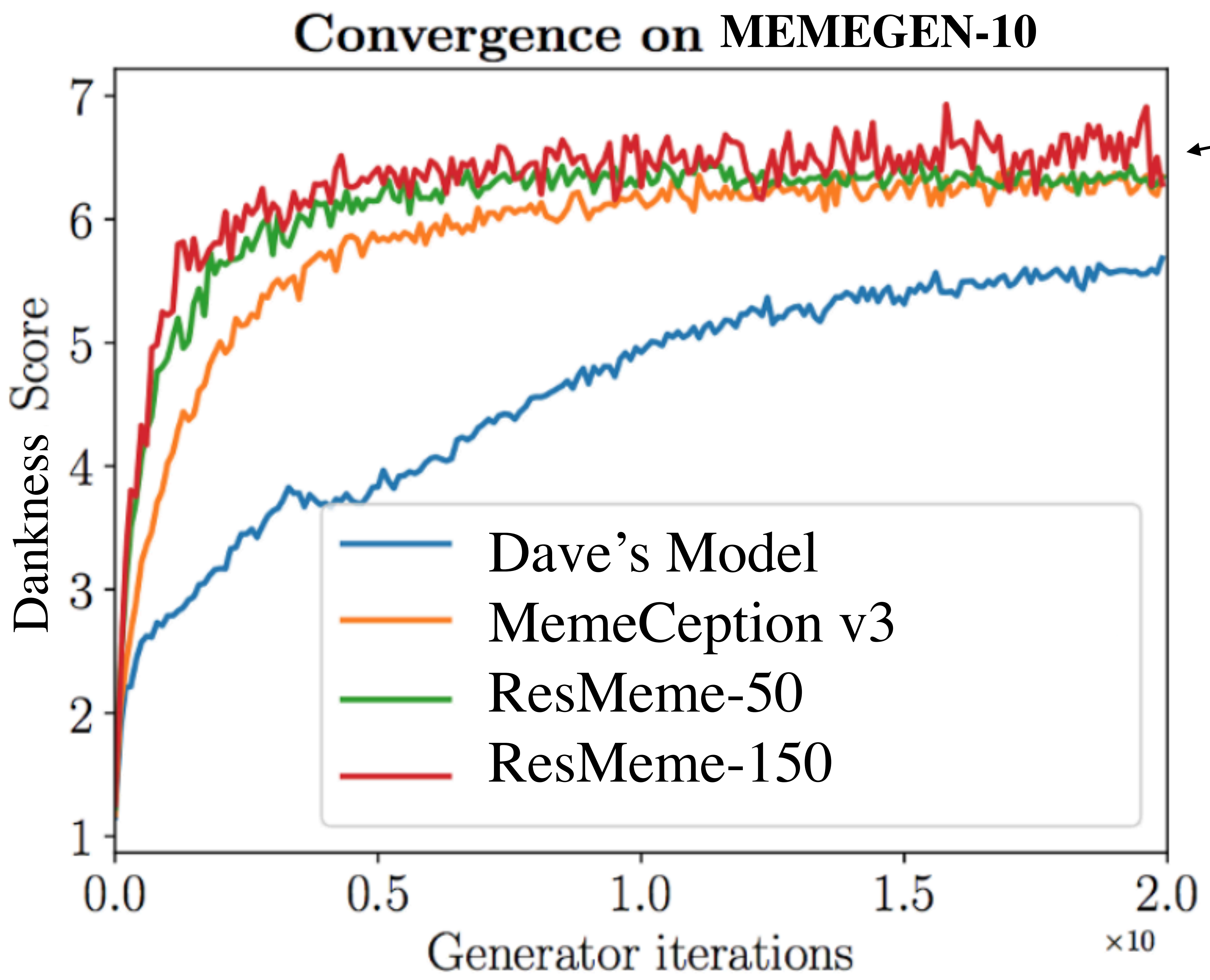
Dave trains his model

```
freshpond:DLRox sparks$ python train_script.py --learning_rate=0.1 --dropout=0.5 > logs/result-0.1-0.5.log
freshpond:DLRox sparks$ ls logs
result-0.1-0.5.log
```

Dave's got a quality problem

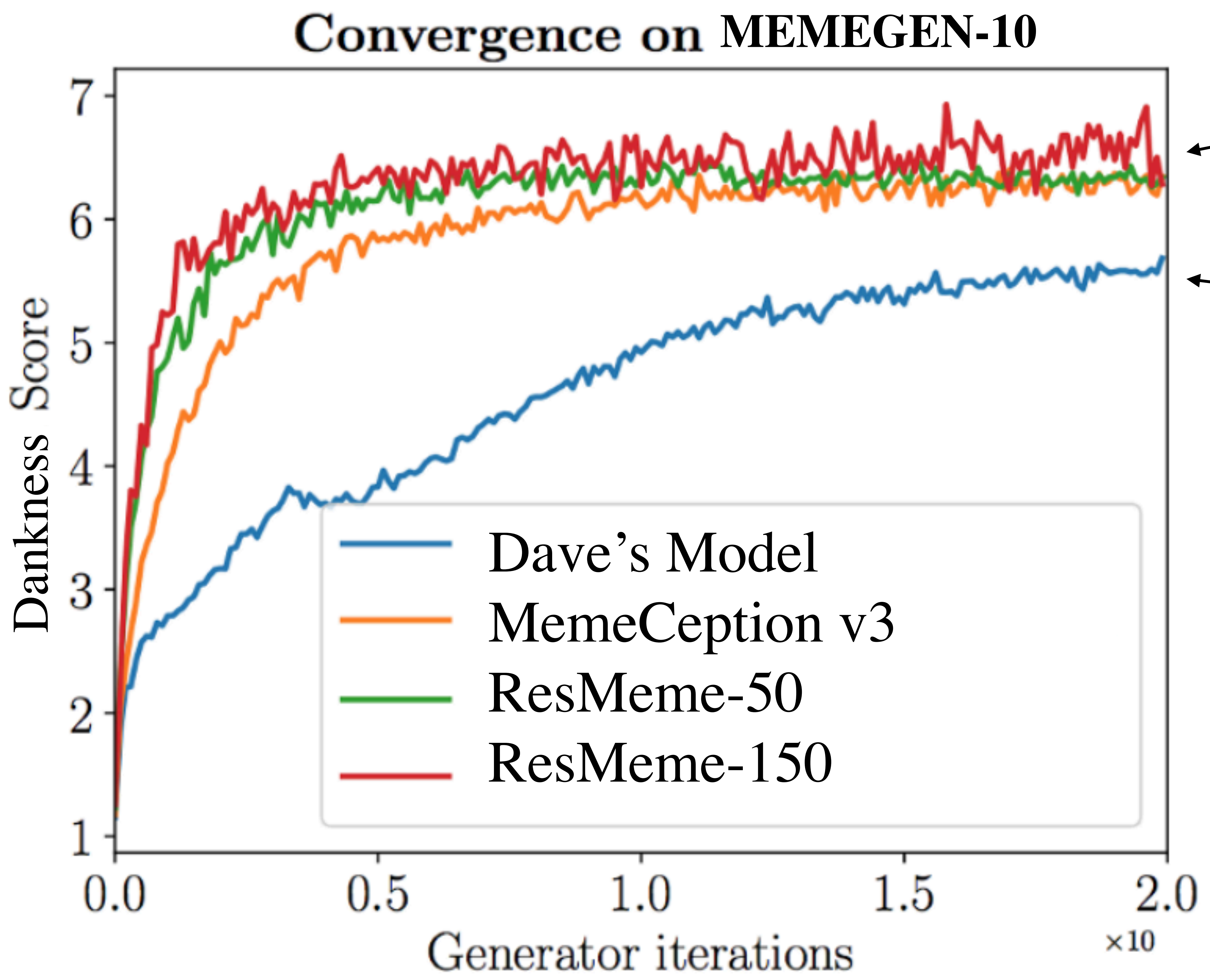


Dave’s got a quality problem



State of the art meme generation.

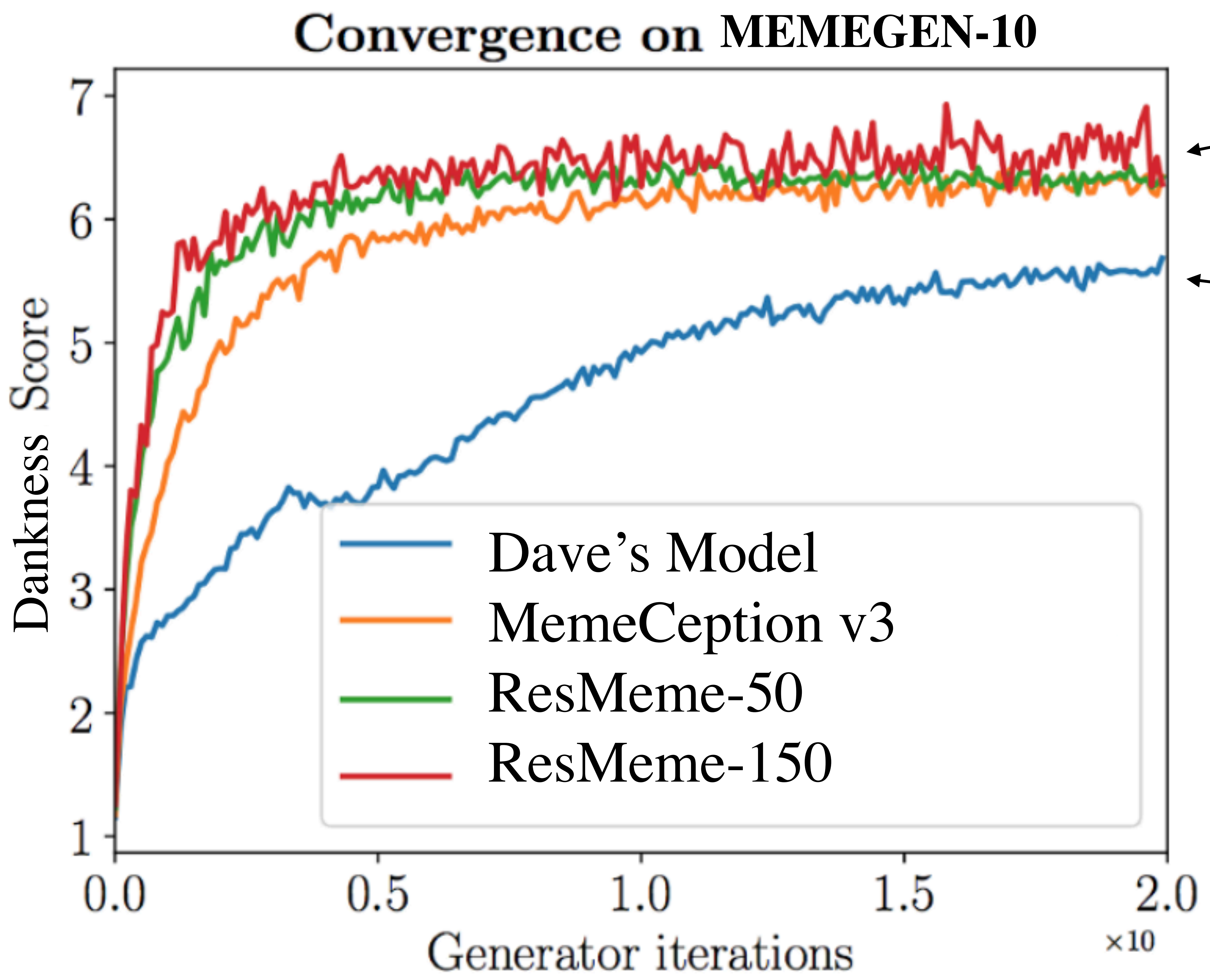
Dave’s got a quality problem



State of the art meme generation.

Dave’s memes

Dave’s got a quality problem

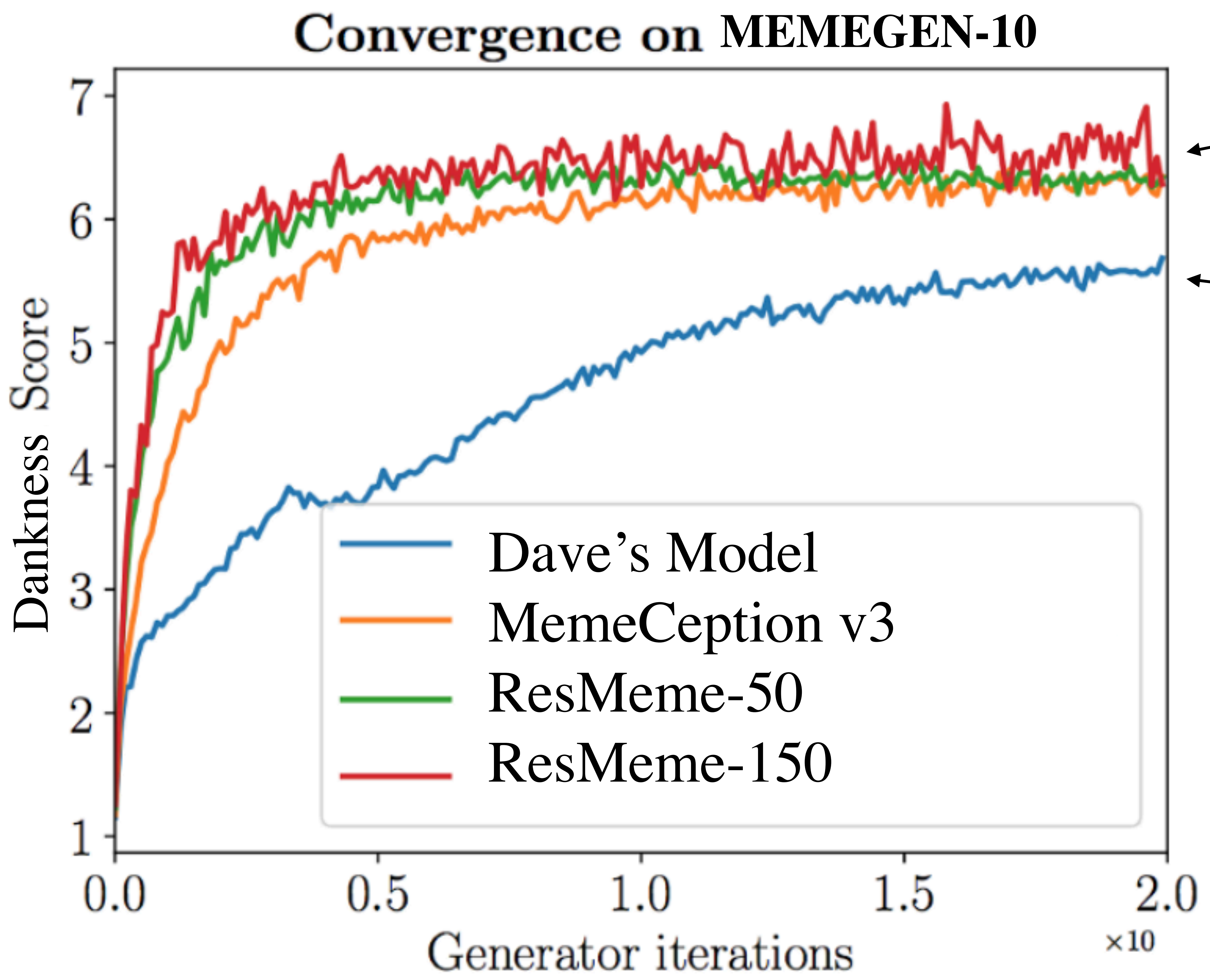


State of the art meme generation.

Dave’s memes

Dave’s memes aren’t dank enough.

Dave's got a quality problem



State of the art meme generation.

Dave's memes

Dave's memes aren't dank enough.

So Dave starts tuning hyperparameters.

Dave Discovers Grid Search

```
freshpond:DLRox sparks$ for lr in -0.001 -0.01 -0.1 1.0 10.0 100.0 1000.0
> do
>   for dropout in 0.0 0.1 0.2 0.3 0.4 0.5
>   do
>     python train_script.py --learning_rate=$lr --dropout=$dropout > logs/results-$lr-$dropout.log
>   done
> done
```

Nested `for` loops FTW

Dave Discovers Grid Search

```
freshpond:DLRox sparks$ ls logs
result-0.1-0.5.log       results--0.01-0.4.log   results-1.0-0.3.log      results-100.0-0.2.log
results--0.001-.log      results--0.01-0.5.log   results-1.0-0.4.log      results-100.0-0.3.log
results--0.001-0.0.log   results--0.1-.log       results-1.0-0.5.log      results-100.0-0.4.log
results--0.001-0.1.log   results--0.1-0.0.log    results-10.0-.log        results-100.0-0.5.log
results--0.001-0.2.log   results--0.1-0.1.log    results-10.0-0.0.log     results-1000.0-.log
results--0.001-0.3.log   results--0.1-0.2.log    results-10.0-0.1.log     results-1000.0-0.0.log
results--0.001-0.4.log   results--0.1-0.3.log    results-10.0-0.2.log     results-1000.0-0.1.log
results--0.001-0.5.log   results--0.1-0.4.log    results-10.0-0.3.log     results-1000.0-0.2.log
results--0.01-.log       results--0.1-0.5.log    results-10.0-0.4.log     results-1000.0-0.3.log
results--0.01-0.0.log    results-1.0-.log        results-10.0-0.5.log     results-1000.0-0.4.log
results--0.01-0.1.log    results-1.0-0.0.log     results-100.0-.log       results-1000.0-0.5.log
results--0.01-0.2.log    results-1.0-0.1.log     results-100.0-0.0.log
results--0.01-0.3.log    results-1.0-0.2.log     results-100.0-0.1.log
```

The results are in.. (kinda)

Dave Discovers Grid Search

Dave Discovers Grid Search

That’s slow, let’s use $CLUSTER_RESOURCE_MANAGER

Dave Discovers Grid Search

That's slow, let's use $CLUSTER_RESOURCE_MANAGER

```
freshpond:DLRox sparks$ for lr in -0.00001 -0.0001 -0.001 -0.01 -0.1 1.0 10.0 100.0 1000.0 10000.0 100000.0 ;
do   for dropout in 0.0 0.05 0.1 0.15 0.2 0.25 0.3 0.35 0.4 0.45 0.5;   do     qsub python train_script.py --l
earning_rate=$lr --dropout=$dropout > logs/results2-$lr-$dropout.log;   done;  done
```

Dave Discovers Grid Search

That’s slow, let’s use $CLUSTER_RESOURCE_MANAGER

```
freshpond:DLRox sparks$ for lr in -0.00001 -0.0001 -0.001 -0.01 -0.1 1.0 10.0 100.0 1000.0 10000.0 100000.0 ;
do   for dropout in 0.0 0.05 0.1 0.15 0.2 0.25 0.3 0.35 0.4 0.45 0.5;   do     qsub python train_script.py --l
earning_rate=$lr --dropout=$dropout > logs/results2-$lr-$dropout.log;   done; done
```

Runs everything in parallel!

Dave Discovers Grid Search

That's slow, let's use $CLUSTER_RESOURCE_MANAGER

```
freshpond:DLRox sparks$ for lr in -0.00001 -0.0001 -0.001 -0.01 -0.1 1.0 10.0 100.0 1000.0 10000.0 100000.0 ;
do   for dropout in 0.0 0.05 0.1 0.15 0.2 0.25 0.3 0.35 0.4 0.45 0.5;   do     qsub python train_script.py --l
earning_rate=$lr --dropout=$dropout > logs/results2-$lr-$dropout.log;   done; done
```

Runs everything in parallel!

No assistance with metadata management, fault tolerance, efficient allocation. AND he's going to throw away 99% of this work!

Dave Discovers Grid Sea

That's slow, let's u RCE_MANAGER

```
freshpond:DLRox sparks$ for lr in -0.0000 1.0 10.0 100.0 1000.0 10000.0 100000.0 ;
do   for dropout in 0.0 0.05 0.1 0.15 0.2 ;   do     qsub python train_script.py --l
earning_rate=$lr --dropout=$dropout > log ;   done;  done
```

```
freshpond:DLRox sparks$ ls logs
result-0.1-0.5.log
results--0.00001-0.0.log
results--0.00001-0.05.log
results--0.00001-0.1.log
results--0.00001-0.15.log
results--0.00001-0.2.log
results--0.00001-0.25.log
results--0.00001-0.3.log
results--0.00001-0.35.log
results--0.00001-0.4.log
results--0.00001-0.45.log
results--0.00001-0.5.log
results--0.0001-0.0.log
results--0.0001-0.05.log
results--0.0001-0.1.log
results--0.0001-0.15.log
results--0.0001-0.2.log
results--0.0001-0.25.log
results--0.0001-0.3.log
results--0.0001-0.35.log
results--0.0001-0.4.log
results--0.0001-0.45.log
results--0.0001-0.5.log
results--0.001-.log
results--0.001-0.0.log
results--0.001-0.05.log
results--0.001-0.1.log
results--0.001-0.15.log
results--0.001-0.2.log
results--0.001-0.25.log
results--0.001-0.3.log
results--0.001-0.35.log
results--0.001-0.4.log
results--0.001-0.45.log
results--0.001-0.5.log
results--0.01-.log
results--0.01-0.0.log
results--0.01-0.05.log
results--0.01-0.1.log
results--0.01-0.15.log
results--0.01-0.2.log
results--0.01-0.25.log
results--0.01-0.3.log
results--0.01-0.35.log
results--0.01-0.4.log
results--0.01-0.45.log
results--0.01-0.5.log
results--0.1-.log
results--0.1-0.0.log
results--0.1-0.05.log
results--0.1-0.1.log
results--0.1-0.15.log
results--0.1-0.2.log
results--0.1-0.25.log
results--0.1-0.3.log
results--0.1-0.35.log
results--0.1-0.4.log
results--0.1-0.45.log
results--0.1-0.5.log
results-1.0-.log
results-1.0-0.0.log
results-1.0-0.05.log
results-1.0-0.1.log
results-1.0-0.15.log
results-1.0-0.2.log
results-1.0-0.25.log
results-1.0-0.3.log
results-1.0-0.35.log
results-1.0-0.4.log
results-1.0-0.45.log
results-1.0-0.5.log
results-10.0-.log
results-10.0-0.0.log
results-10.0-0.05.log
results-10.0-0.1.log
results-10.0-0.15.log
results-10.0-0.2.log
results-10.0-0.25.log
results-10.0-0.3.log
results-10.0-0.35.log
results-10.0-0.4.log
results-10.0-0.45.log
results-10.0-0.5.log
results-100.0-0.0.log
results-100.0-0.05.log
results-100.0-0.1.log
results-100.0-0.15.log
results-100.0-0.2.log
results-100.0-0.25.log
results-100.0-0.3.log
results-100.0-0.35.log
results-100.0-0.4.log
results-100.0-0.45.log
results-100.0-0.5.log
results-1000.0-.log
results-1000.0-0.0.log
results-1000.0-0.05.log
results-1000.0-0.1.log
results-1000.0-0.15.log
results-1000.0-0.2.log
results-1000.0-0.25.log
results-1000.0-0.3.log
results-1000.0-0.35.log
results-1000.0-0.4.log
results-1000.0-0.45.log
results-1000.0-0.5.log
results-10000.0-0.0.log
results-10000.0-0.05.log
results-10000.0-0.1.log
results-10000.0-0.15.log
results-10000.0-0.2.log
results-10000.0-0.25.log
results-10000.0-0.3.log
results-10000.0-0.35.log
results-10000.0-0.4.log
results-10000.0-0.45.log
results-10000.0-0.5.log
results-100000.0-0.0.log
results-100000.0-0.05.log
results-100000.0-0.1.log
results-100000.0-0.15.log
results-100000.0-0.2.log
results-100000.0-0.25.log
results-100000.0-0.3.log
results-100000.0-0.35.log
results-100000.0-0.4.log
results-100000.0-0.45.log
results-100000.0-0.5.log
results2--0.00001-0.0.log
results2--0.00001-0.05.log
results2--0.00001-0.1.log
results2--0.00001-0.15.log
results2--0.00001-0.2.log
results2--0.00001-0.25.log
results2--0.00001-0.3.log
results2--0.00001-0.35.log
results2--0.00001-0.4.log
results2--0.00001-0.45.log
results2--0.00001-0.5.log
results2--0.0001-0.0.log
results2--0.0001-0.05.log
results2--0.0001-0.1.log
results2--0.0001-0.15.log
results2--0.0001-0.2.log
results2--0.0001-0.25.log
results2--0.0001-0.3.log
results2--0.0001-0.35.log
results2--0.0001-0.4.log
results2--0.0001-0.45.log
results2--0.0001-0.5.log
results2--0.001-0.0.log
results2--0.001-0.05.log
results2--0.001-0.1.log
results2--0.001-0.15.log
results2--0.001-0.2.log
results2--0.001-0.25.log
results2--0.001-0.3.log
results2--0.001-0.35.log
results2--0.001-0.4.log
results2--0.001-0.45.log
results2--0.001-0.5.log
results2--0.01-0.0.log
results2--0.01-0.05.log
results2--0.01-0.1.log
results2--0.01-0.15.log
results2--0.01-0.2.log
results2--0.01-0.3.log
results2--0.01-0.35.log
results2--0.01-0.4.log
results2--0.01-0.45.log
results2--0.01-0.5.log
results2--0.1-0.0.log
results2--0.1-0.05.log
results2--0.1-0.1.log
results2--0.1-0.15.log
results2--0.1-0.2.log
results2--0.1-0.25.log
results2--0.1-0.3.log
results2--0.1-0.35.log
results2--0.1-0.4.log
results2--0.1-0.45.log
results2--0.1-0.5.log
results2-1.0-0.0.log
results2-1.0-0.05.log
results2-1.0-0.1.log
results2-1.0-0.15.log
results2-1.0-0.2.log
results2-1.0-0.25.log
results2-1.0-0.3.log
results2-1.0-0.35.log
results2-1.0-0.4.log
results2-1.0-0.45.log
results2-1.0-0.5.log
results2-10.0-0.0.log
results2-10.0-0.05.log
results2-10.0-0.1.log
results2-10.0-0.15.log
results2-10.0-0.2.log
results2-10.0-0.25.log
results2-10.0-0.3.log
results2-10.0-0.35.log
results2-10.0-0.4.log
results2-10.0-0.45.log
results2-10.0-0.5.log
results2-100.0-0.0.log
results2-100.0-0.05.log
results2-100.0-0.1.log
results2-100.0-0.15.log
results2-100.0-0.2.log
results2-100.0-0.25.log
results2-100.0-0.3.log
results2-100.0-0.35.log
results2-100.0-0.4.log
results2-100.0-0.45.log
results2-100.0-0.5.log
results2-1000.0-0.0.log
results2-1000.0-0.05.log
results2-1000.0-0.1.log
results2-1000.0-0.15.log
results2-1000.0-0.2.log
results2-1000.0-0.25.log
results2-1000.0-0.3.log
results2-1000.0-0.35.log
results2-1000.0-0.4.log
results2-1000.0-0.45.log
results2-1000.0-0.5.log
results2-10000.0-0.0.log
results2-10000.0-0.05.log
results2-10000.0-0.1.log
results2-10000.0-0.15.log
results2-10000.0-0.2.log
results2-10000.0-0.25.log
results2-10000.0-0.3.log
results2-10000.0-0.35.log
results2-10000.0-0.4.log
results2-10000.0-0.45.log
results2-10000.0-0.5.log
results2-100000.0-0.0.log
results2-100000.0-0.05.log
results2-100000.0-0.1.log
results2-100000.0-0.15.log
results2-100000.0-0.2.log
results2-100000.0-0.25.log
results2-100000.0-0.3.log
results2-100000.0-0.35.log
results2-100000.0-0.4.log
results2-100000.0-0.45.log
results2-100000.0-0.5.log
```

Ru el!

No assistance with metadata ma nce, efficient allocation. AND he's going t is work!

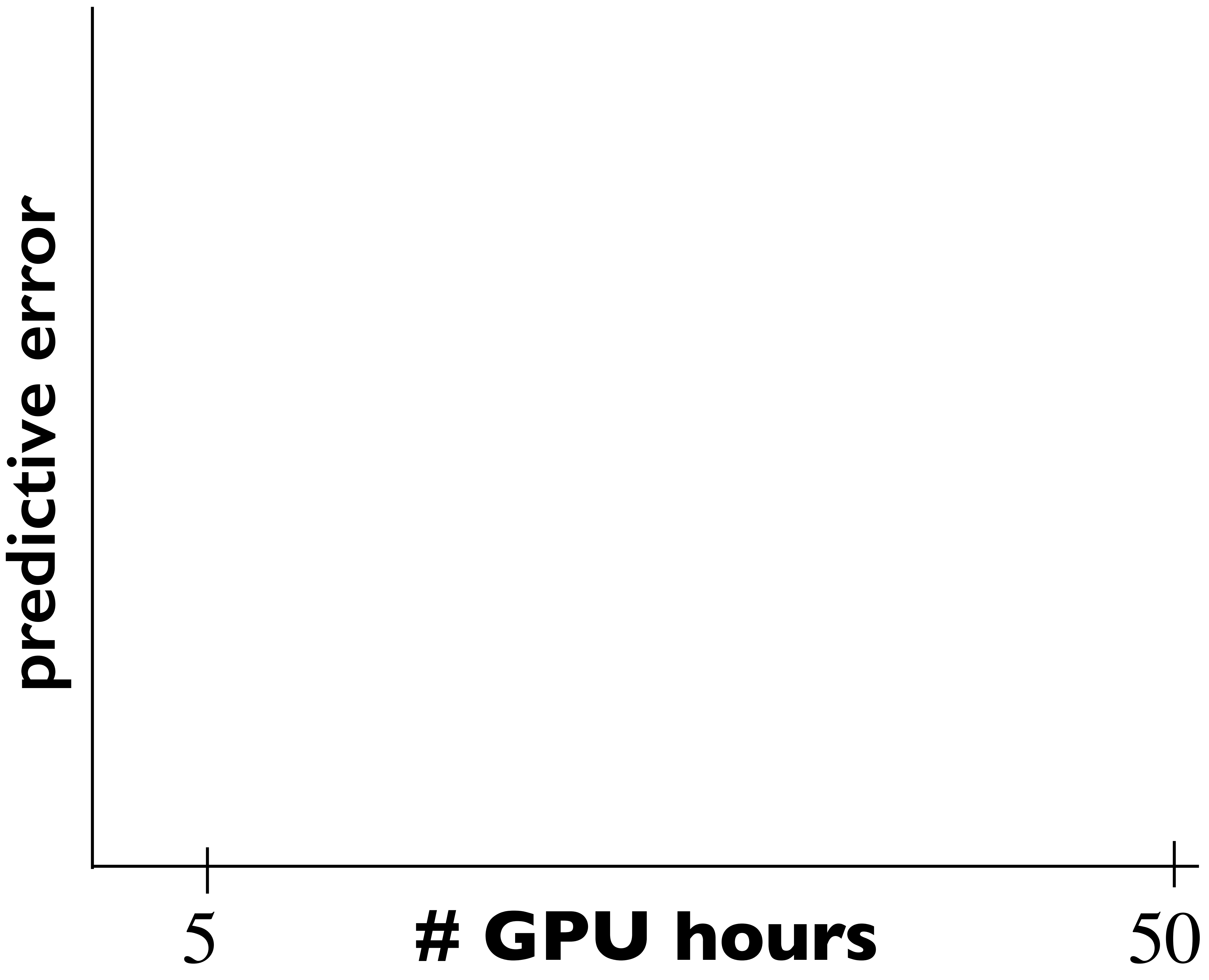
Hyperband: Resource-optimized HPO

4 layer CNN

8 Hyperparameters

Image recognition

CIFAR10



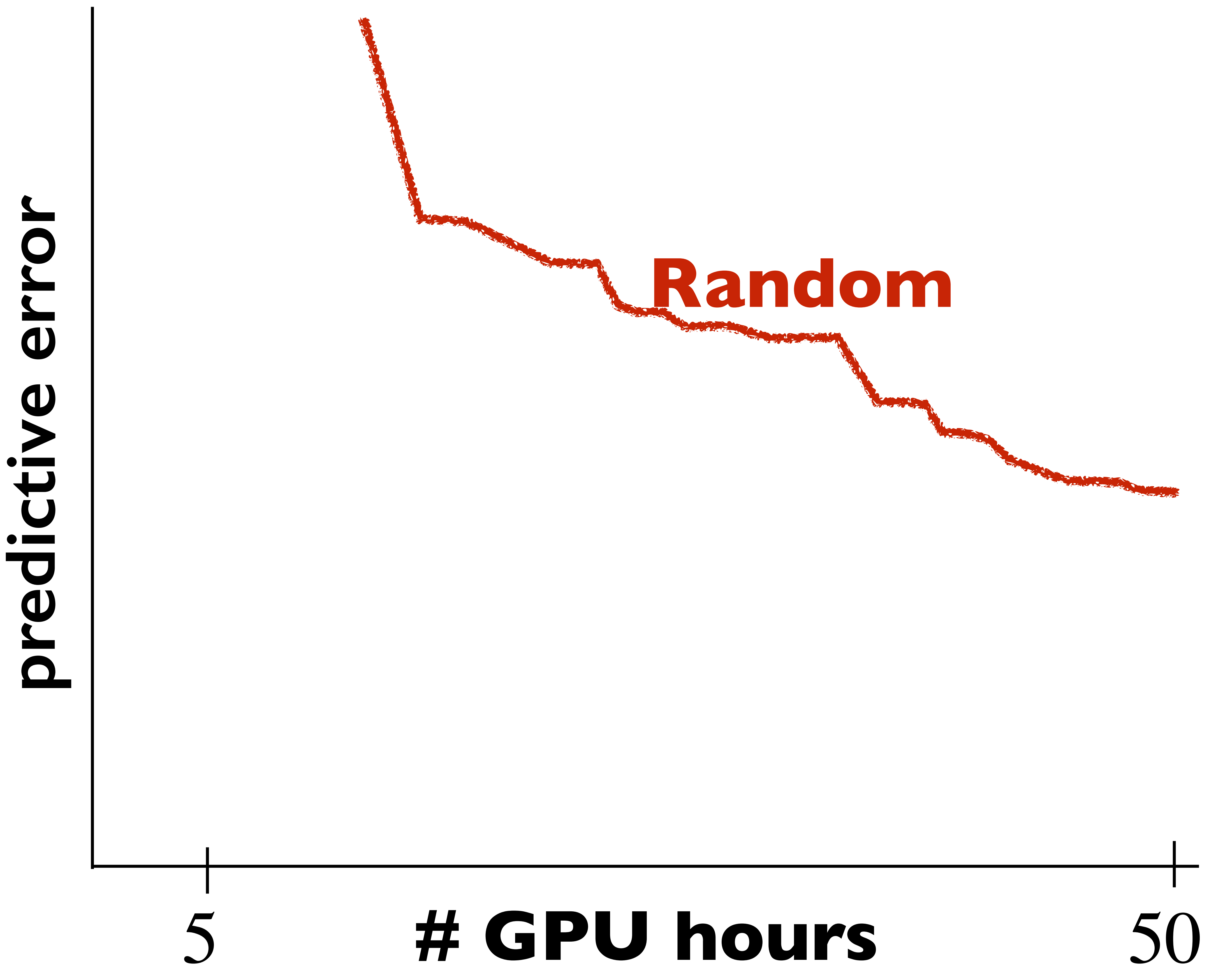
Hyperband: Resource-optimized HPO

4 layer CNN

8 Hyperparameters

Image recognition

CIFAR10



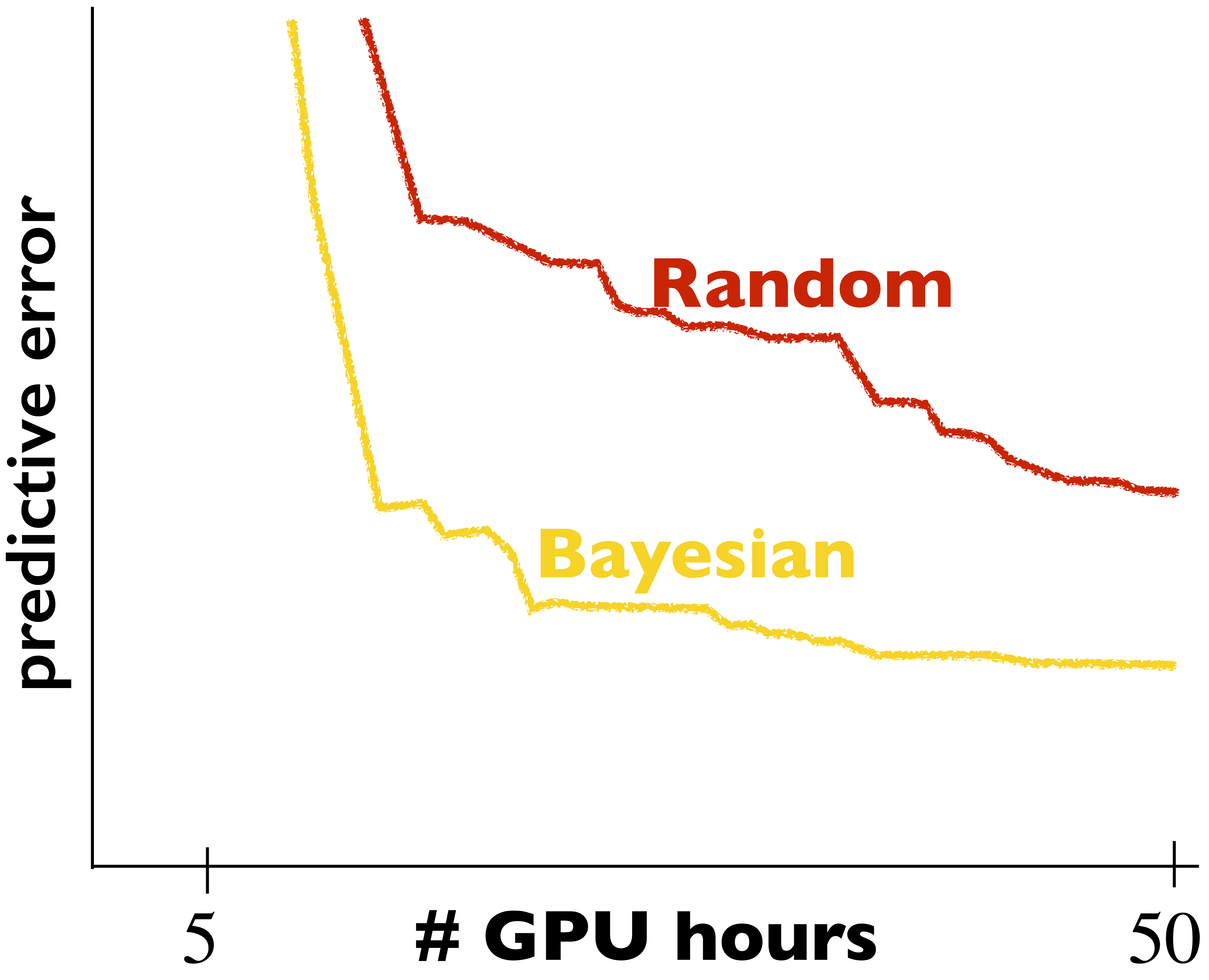
Hyperband: Resource-optimized HPO

4 layer CNN

8 Hyperparameters

Image recognition

CIFAR10



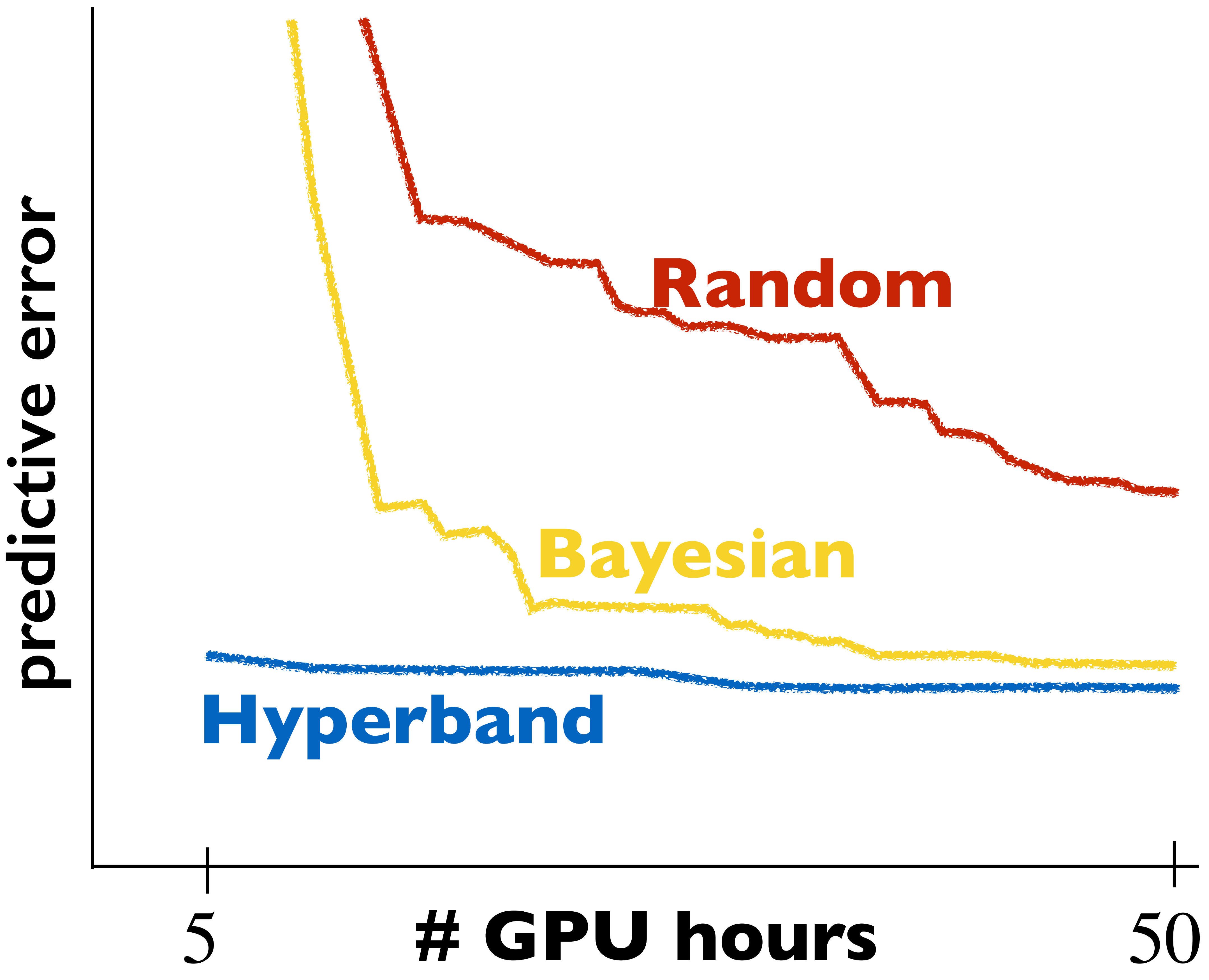
Hyperband: Resource-optimized HPO

4 layer CNN

8 Hyperparameters

Image recognition

CIFAR10



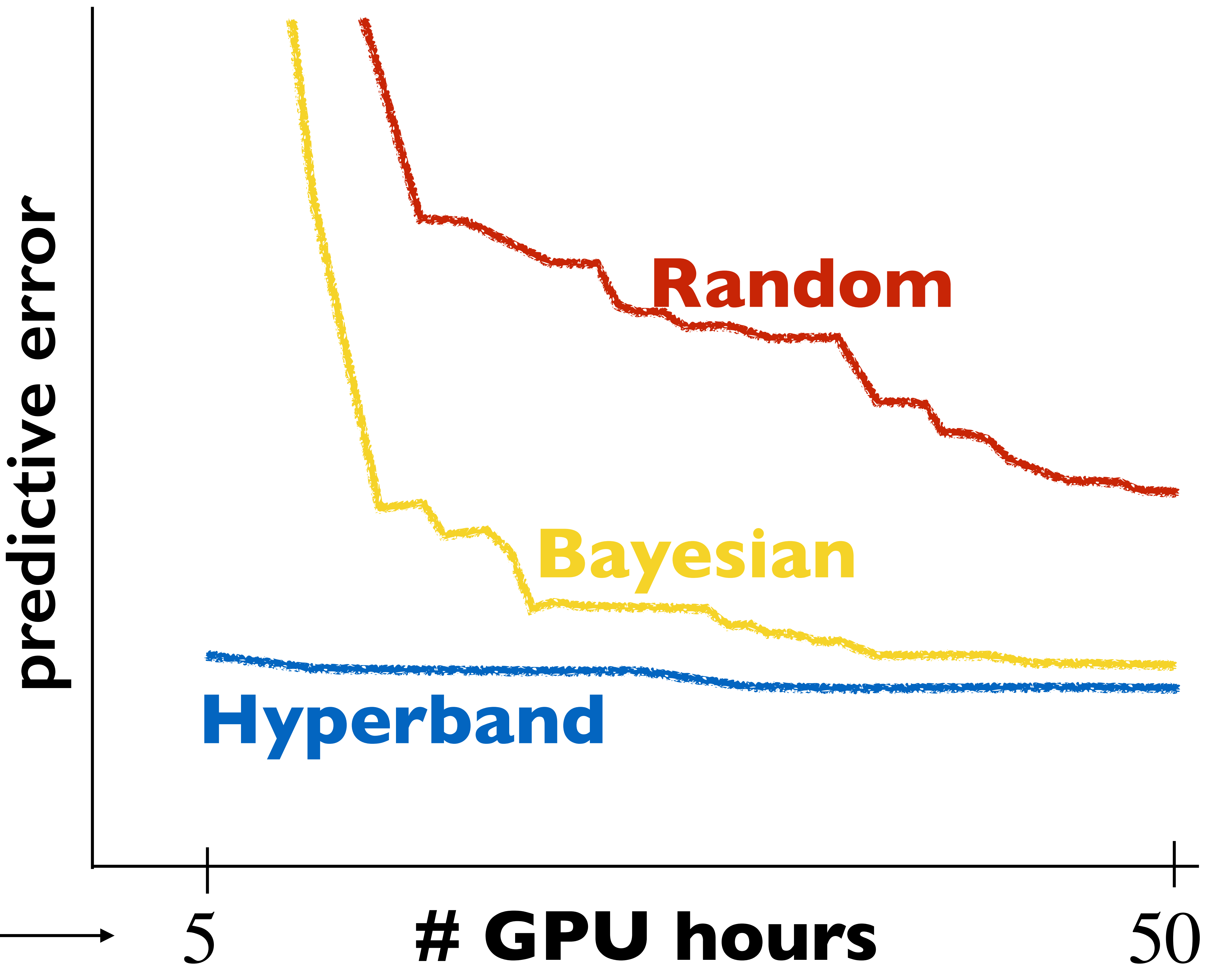
Hyperband: Resource-optimized HPO

4 layer CNN

8 Hyperparameters

Image recognition

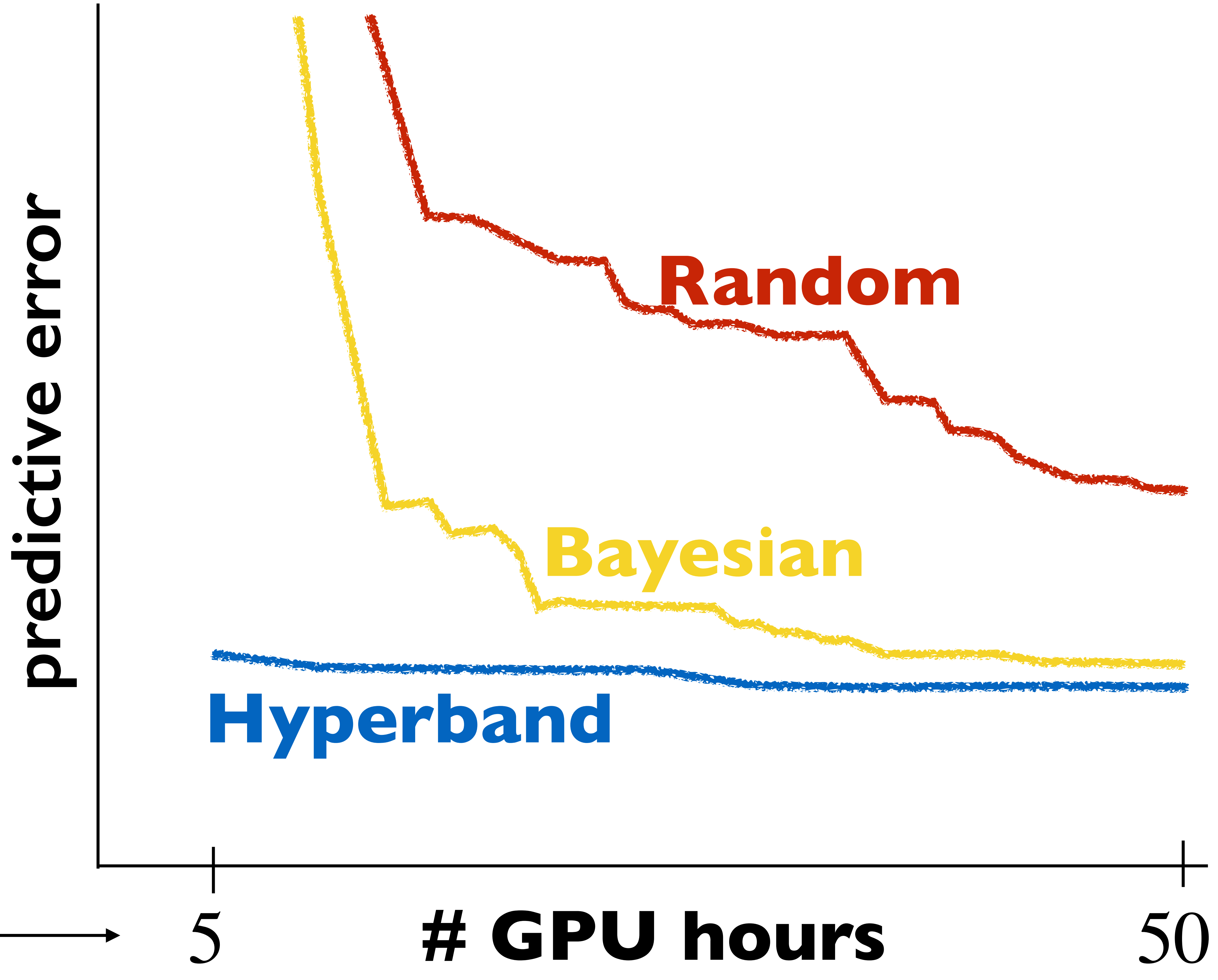
CIFAR10



Hyperband has considered 256 configurations!

Hyperband: Resource-optimized HPO

Speedups

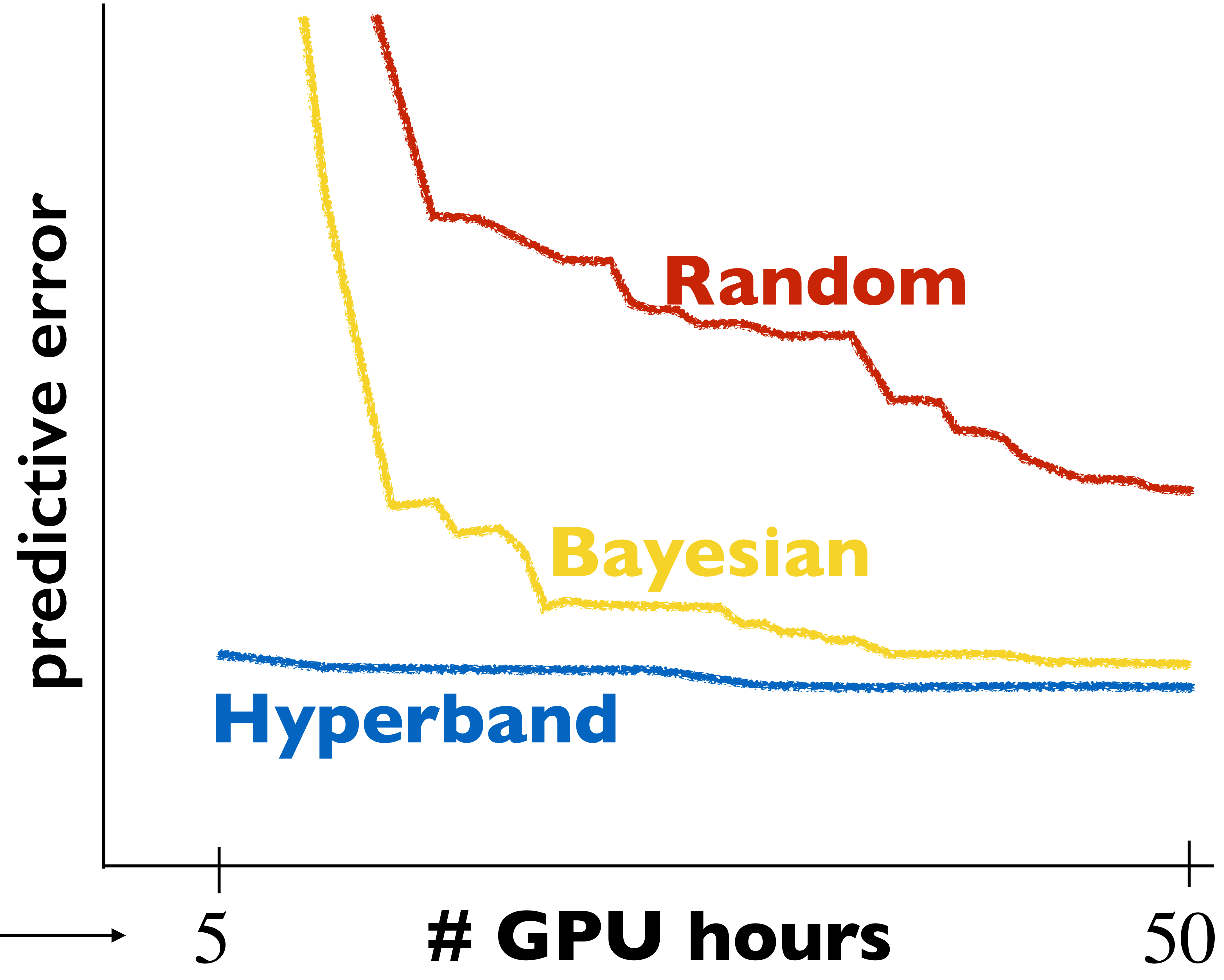


Hyperband has considered 256 configurations!

Hyperband: Resource-optimized HPO

Speedups

>50x over Random



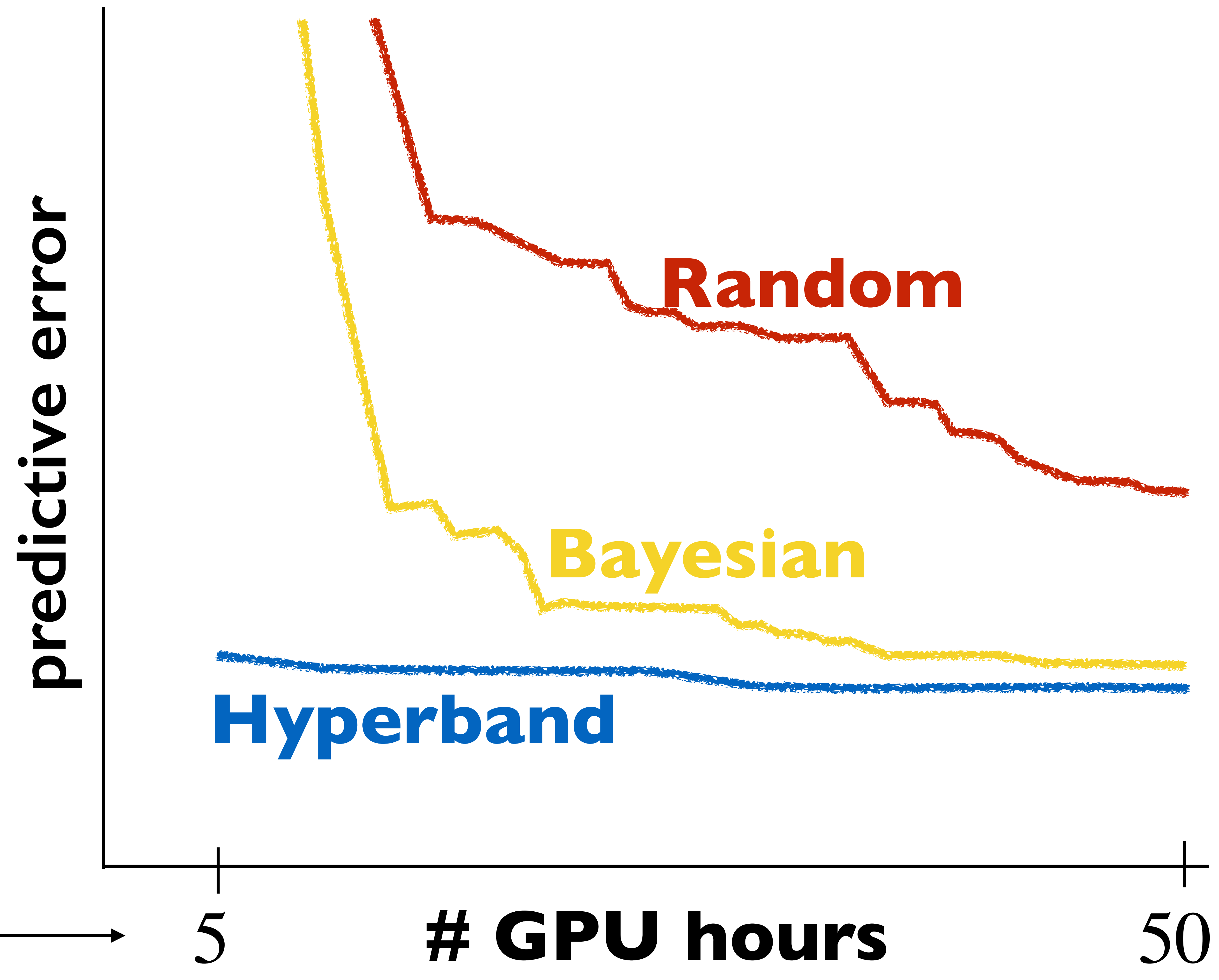
Hyperband has considered 256 configurations!

Hyperband: Resource-optimized HPO

Speedups

>50x over Random

10x over Bayesian



Hyperband has considered 256 configurations!

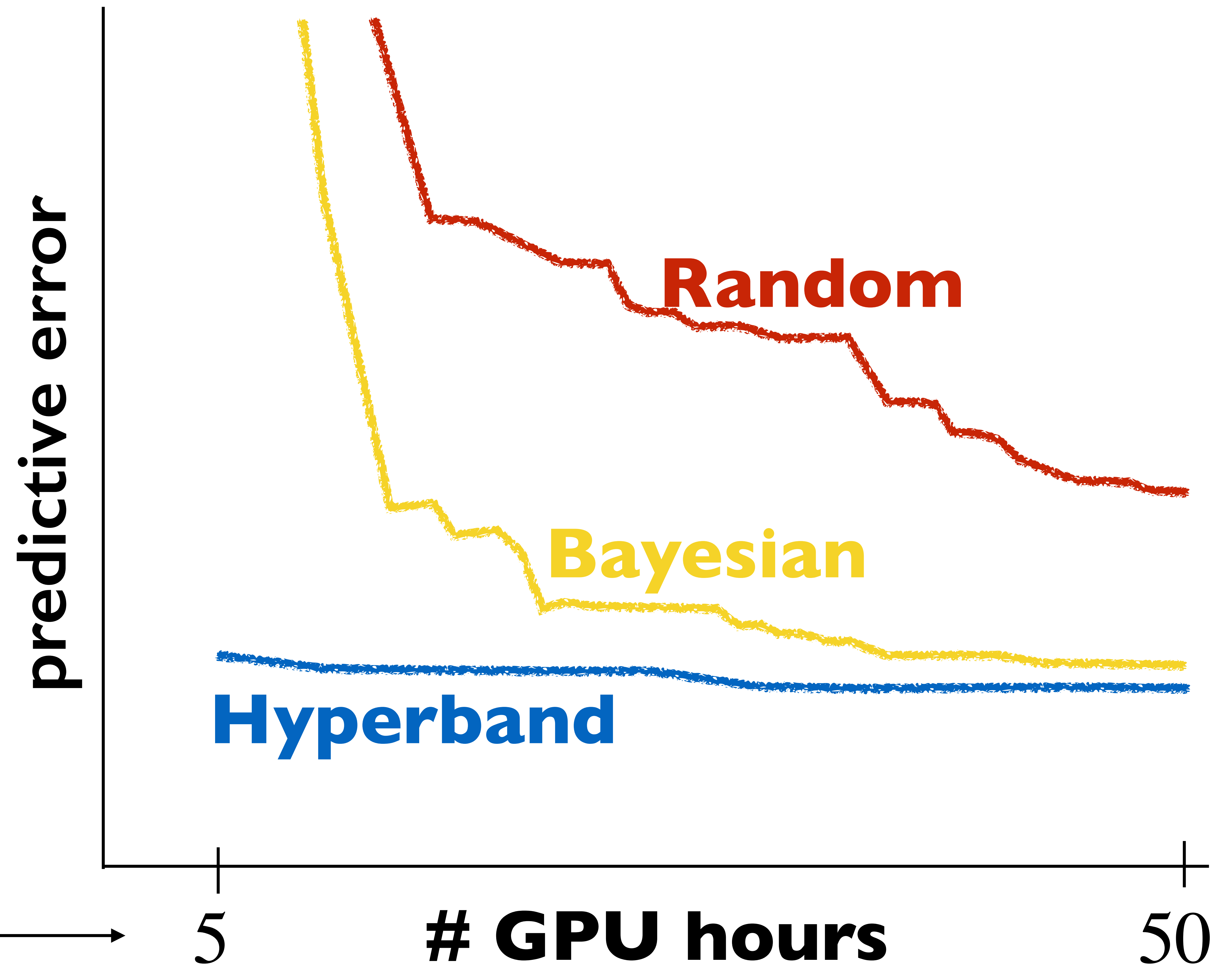
Hyperband: Resource-optimized HPO

Speedups

>50x over Random

10x over Bayesian

✔ **Lower final error**



Hyperband has considered 256 configurations! ⟶

Hyperband: Resource-optimized HPO

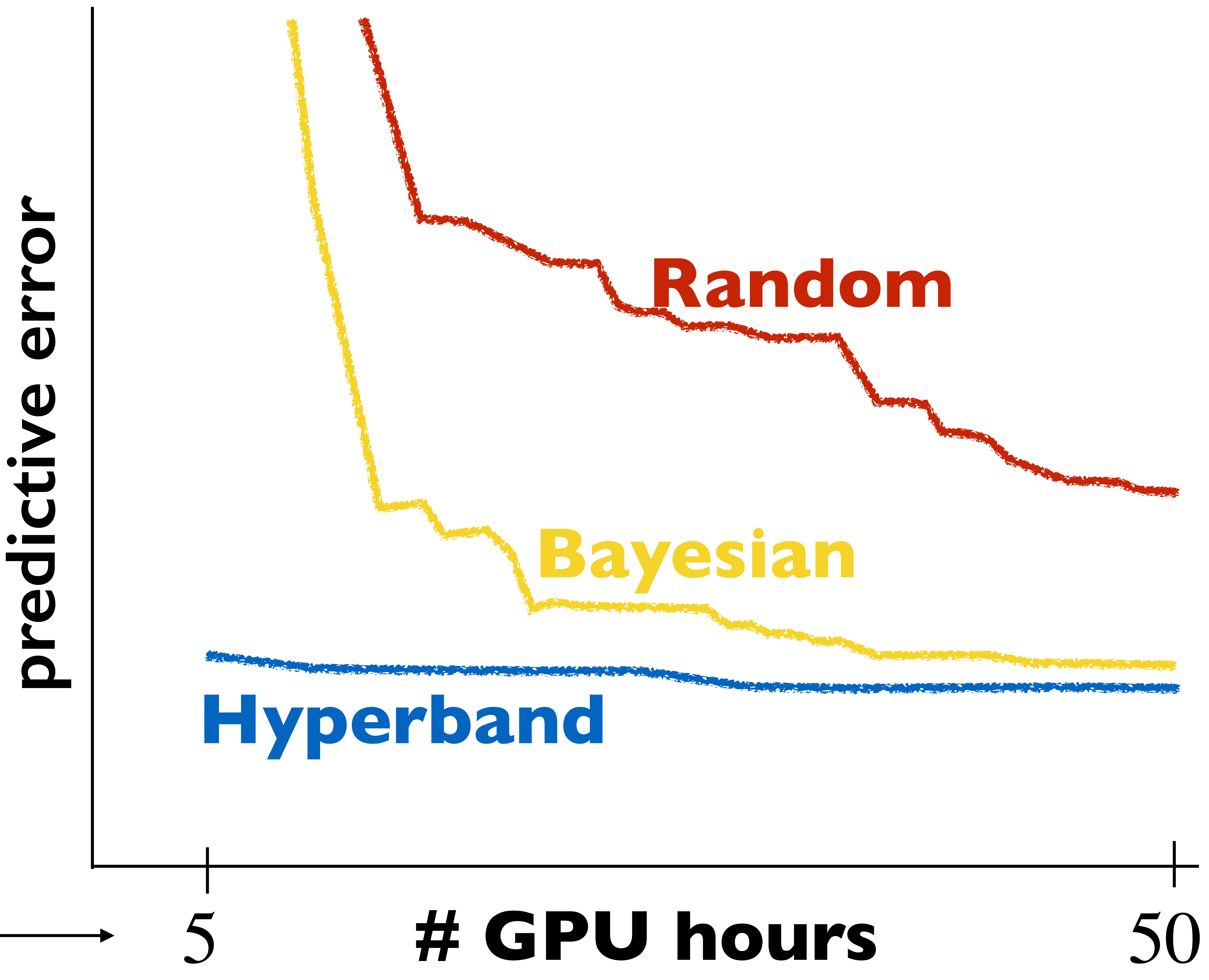
Speedups

>50x over Random

10x over Bayesian

✔ **Lower final error**

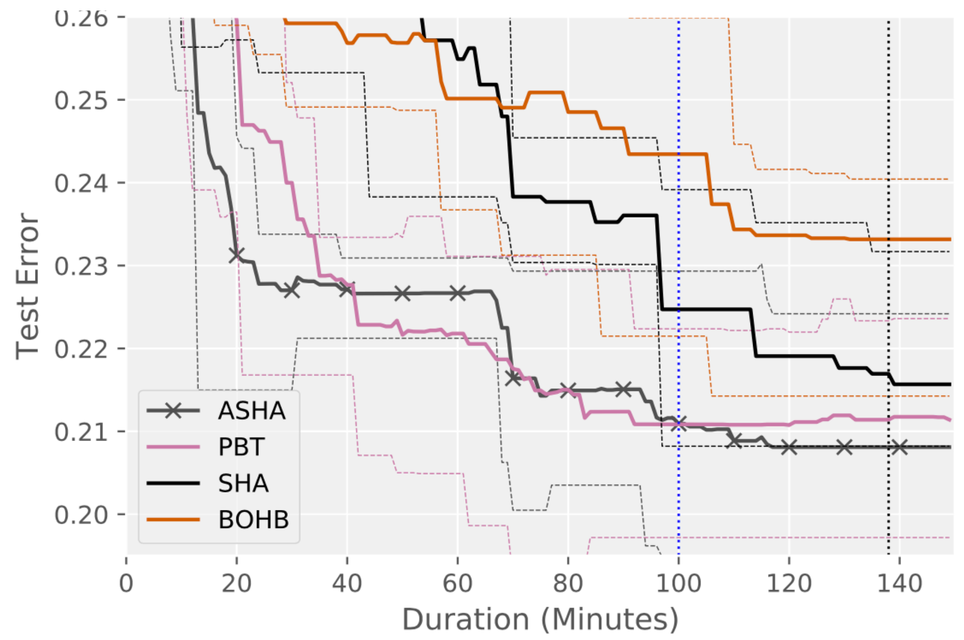
✔ **Lower variance**



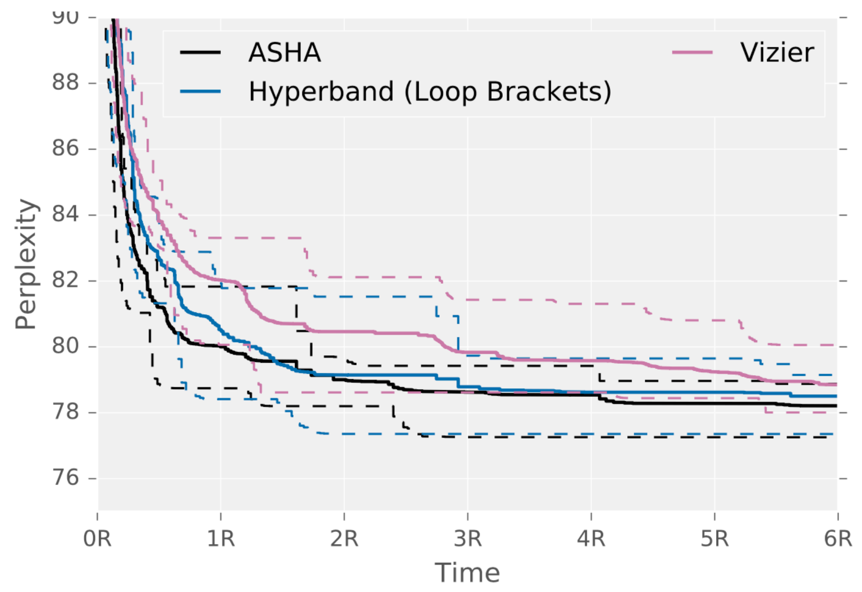
Hyperband has considered 256 configurations! →

Hyperband: Also great at NAS

CIFAR10, CNN, d=10, 25 workers

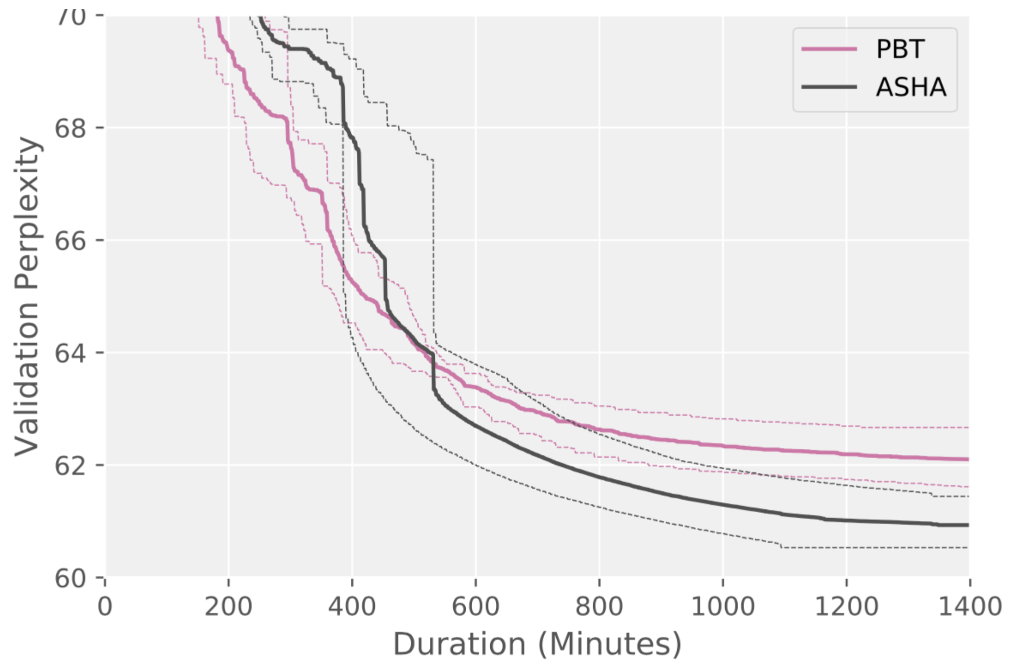


Penn Treebank, LSTM, d=9, 500 workers



Experiment performed in TensorFlow @ Google

Penn Treebank, LSTM, d=9, 16 workers



Penn Treebank, LSTM, d=9, 16 workers

Model	Size	Depth	Valid	Test
Medium LSTM, Zaremba et al. (2014)	10M	2	86.2	82.7
Large LSTM, Zaremba et al. (2014)	24M	2	82.2	78.4
VD LSTM, Press & Wolf (2016)	51M	2	75.8	73.2
VD LSTM, Inan et al. (2016)	9M	2	77.1	73.9
VD LSTM, Inan et al. (2016)	28M	2	72.5	69.0
VD RHN, Zilly et al. (2016)	24M	10	67.9	65.4
NAS, Zoph & Le (2016)	25M	-	-	64.0
NAS, Zoph & Le (2016)	54M	-	-	62.4
AWD-LSTM, Merity et al. (2017) †	24M	3	60.0	57.3
ASH	24	3	58.	56.

Unfortunately, Dave can’t Hyperband

Dave's Infrastructure Dilemma

Cluster Manager:

Doesn't understand the semantics of deep learning

DL Frameworks:

Built to train a single model for a single user on a single machine

What's missing is **holistic** but **specialized** infrastructure to provide the glue between these two

The Dark Age of AI Infrastructure

Forcing users to wait for **days** to recover from faults.

Reproducing existing models is **death by a thousand cuts:** data ordering, software versions, hyperaparmeters, random seeds, model weights.

Hand-implemented, **impossibly slow** methods to find good models.

Trapping our users in systems designed to house **one user** with **rigid infrastructure.**

The Dark Age of AI Infrastructure

Forcing users to wait for **days** to recover from faults.

Reproducing existing models is **death by a thousand cuts:** data ordering, software versions, hyperaparmeters, random seeds, model weights.

Hand-implemented, **impossibly slow** methods to find good models.

Trapping our users in systems designed to house **one user** with **rigid infrastructure.**

Why Should Dave Care?

Scientific Progress

Collaboration

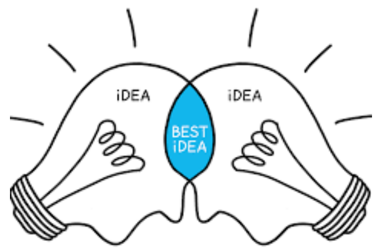
Accountability

Reproducibility is a fundamental tenet of scientific progress

Hidden sources of randomness can lead to erroneous conclusions

Why Should Dave Care?

Scientific Progress



Collaboration

Accountability

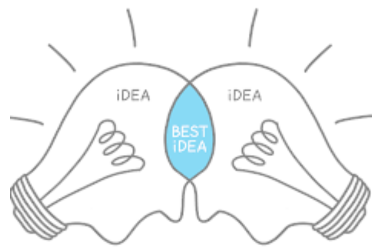
Enable sharing & encourages experimentation

Easily ramp-up new hires

Reduce dependency on individual team member

Why Should Dave Care?

Scientific Progress



Collaboration

Accountability

Avoid lossy translation between training and deployment

Easily roll back in case of system crash or poor performance

Wait…isn't this a solved problem?

Wait…isn't this a solved problem?

Traditional Software Engineering

compile(code, deps) → binary

Wait…isn’t this a solved problem?

Traditional Software Engineering

compile(code, deps) → binary

Deep Learning Engineering

optimize(architecture, deps, data, init state) → ML model

Wait…isn't this a solved problem?

Traditional Software Engineering

compile(code, deps) → binary

Deep Learning Engineering

optimize(architecture, deps, data, init state) → ML model

Additional inputs + noisy optimizer = **ML reproducibility is hard!**

Dave is taking over for Leslie

He re-runs Leslie's training script but get **drastically higher error**

Time to debug…

What does Dave discover?

Training data: New samples recently added to Leslie's directory

Hyperparameters: Leslie didn't use default values, and instead specified batch size and learning rate at runtime

What does Dave discover?

Training data: New samples recently added to Leslie's directory

Hyperparameters: Leslie didn't use default values, and instead specified batch size and learning rate at runtime

	Validation Error	Difference from Baseline
Baseline	30.3%	0.0%
Test1 (w/o fixes)	52.8%	22.5%

What does Dave discover?

Training data: New samples recently added to Leslie's directory

Hyperparameters: Leslie didn't use default values, and instead specified batch size and learning rate at runtime

	Validation Error	Difference from Baseline
Baseline	30.3%	0.0%
Test1 (w/o fixes)	52.8%	22.5%
Test2 (includes fixes)	37.3%	**7%**

Ugh...Debug...

Ugh…Debug…

Randomness is an intrinsic part of training

- e.g., weight initialization, shuffling and augmentation of datasets, noisy hidden layers (e.g. dropout)

Ugh...Debug...

Randomness is an intrinsic part of training

- e.g., weight initialization, shuffling and augmentation of datasets, noisy hidden layers (e.g. dropout)

Fix random seeds!

Ugh...Debug...

Randomness is an intrinsic part of training

- e.g., weight initialization, shuffling and augmentation of datasets, noisy hidden layers (e.g. dropout)

Fix random seeds!

- There are lots of them!
- ML framework dependent
- Must be recorded for reuse

Ugh…Debug…

Ugh…Debug…

Variation across specialized software

- Within versions and across ML frameworks (TF, Keras, PyTorch)
- Underlying libraries (NumPy, cuDNN, CUDA, MKL)

Ugh…Debug…

Variation across specialized software

- Within versions and across ML frameworks (TF, Keras, PyTorch)
- Underlying libraries (NumPy, cuDNN, CUDA, MKL)

Leverage the power of containerization!

Ugh...Debug...

Variation across specialized software

- Within versions and across ML frameworks (TF, Keras, PyTorch)
- Underlying libraries (NumPy, cuDNN, CUDA, MKL)

Leverage the power of containerization!

Requires non-trivial engineering infrastructure

Ugh...Debug...

Ugh...Debug...

	Validation Error	Difference from Baseline
Baseline	30.3%	0.0%
Test1: No changes	52.8%	22.5%
Test2: Fix dataset + hyperparameters	37.3%	7.0%

Ugh...Debug...

	Validation Error	Difference from Baseline
Baseline	30.3%	0.0%
Test1: No changes	52.8%	22.5%
Test2: Fix dataset + hyperparameters	37.3%	7.0%
Test3: Fix for libraries + random seeds	29.2%	-1.1%

Ugh...Debug...

	Validation Error	Difference from Baseline
Baseline	30.3%	0.0%
Test1: No changes	52.8%	22.5%
Test2: Fix dataset + hyperparameters	37.3%	7.0%
Test3: Fix for libraries + random seeds	29.2%	**-1.1%**

UGH!!!

Inherent System/Hardware Level Randomness

- non-deterministic GPU operations
- CPU multi-threading

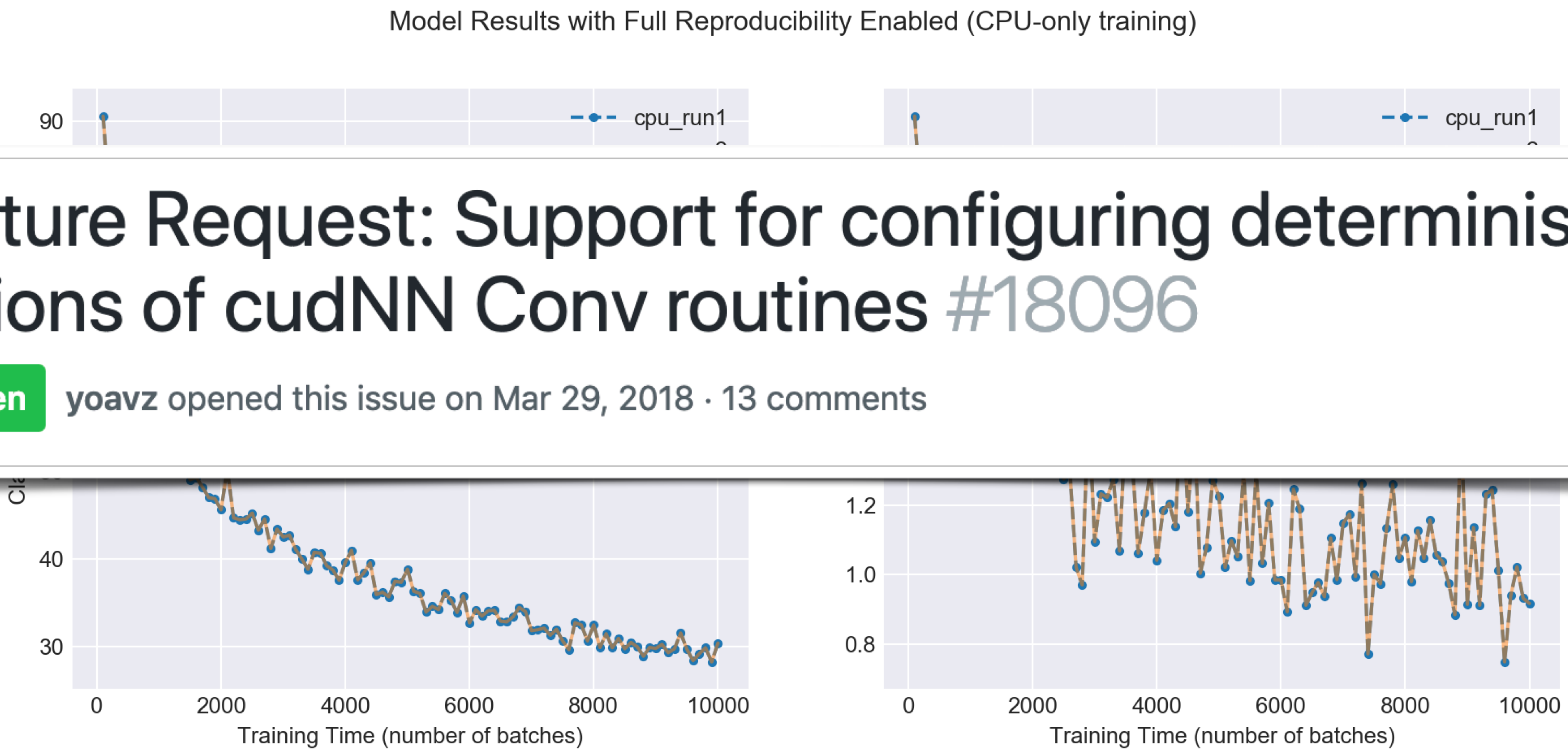
Fixing this, at last, we have perfect reproducibility

Fixing this, at last, we have perfect reproducibility

Fixing this, at last, we have perfect reproducibility

But it requires **CPU-only** training with multi-threading disabled…**SLOW**!

Fixing this, at last, we have perfect reproducibility



Feature Request: Support for configuring deterministic options of cudNN Conv routines #18096

Open **yoavz** opened this issue on Mar 29, 2018 · 13 comments

But it requires **CPU-only** training with multi-threading disabled…**SLOW**!

What would an <u>holistic</u> but <u>specialized</u> DL reproducibility solution include?

Feature	Purpose
Version control for model definitions	Track changes in model architecture, optimization algorithm, data preprocessing pipeline
Metadata capture and storage	Record training + validation metrics, training logs, model hyperparameters
Dependency management	Ensure ML framework and all dependencies are consistent between runs
Experiment seed management	Generate the same pseudo-random values every run
Hardware resource flexibility	Allow users to disable multi-threading and GPU usage, if desired

The Dark Age of AI Infrastructure

Forcing users to wait for **days** to recover from faults.

Reproducing existing models is **death by a thousand cuts:** data ordering, software versions, hyperaparmeters, random seeds, model weights.

Hand-implemented, **impossibly slow** methods to find good models.

Trapping our users in systems designed to house **one user** with **rigid infrastructure.**

The Dark Age of AI Infrastructure

Forcing users to wait for **days** to recover from faults.

Reproducing existing models is **death by a thousand cuts:** data ordering, software versions, hyperaparmeters, random seeds, model weights.

Hand-implemented, **impossibly slow** methods to find good models.

Trapping our users in systems designed to house **one user** with **rigid infrastructure.**

Dave gets a new teammate.

Fixed Schedule: Dave gets GPUs on Monday, Leslie on Tuesday

Dedicated Assignment: Dave gets some GPUs, Leslie gets the rest

Calendar Signup: Dave and Leslie share via spreadsheet signups

Sharing is Hard!

Fixed Schedule: Dave gets GPUs on Monday, Leslie on Tuesday

Dedicated Assignment: Dave gets some GPUs, Leslie gets the rest

Calendar Signup: Dave and Leslie share via spreadsheet signups

Sharing is Hard!

Fixed Schedule: Dave gets GPUs on Monday, Leslie on Tuesday

Dedicated Assignment: Dave gets some GPUs, Leslie gets the rest

Calendar Signup: Dave and Leslie share via spreadsheet signups

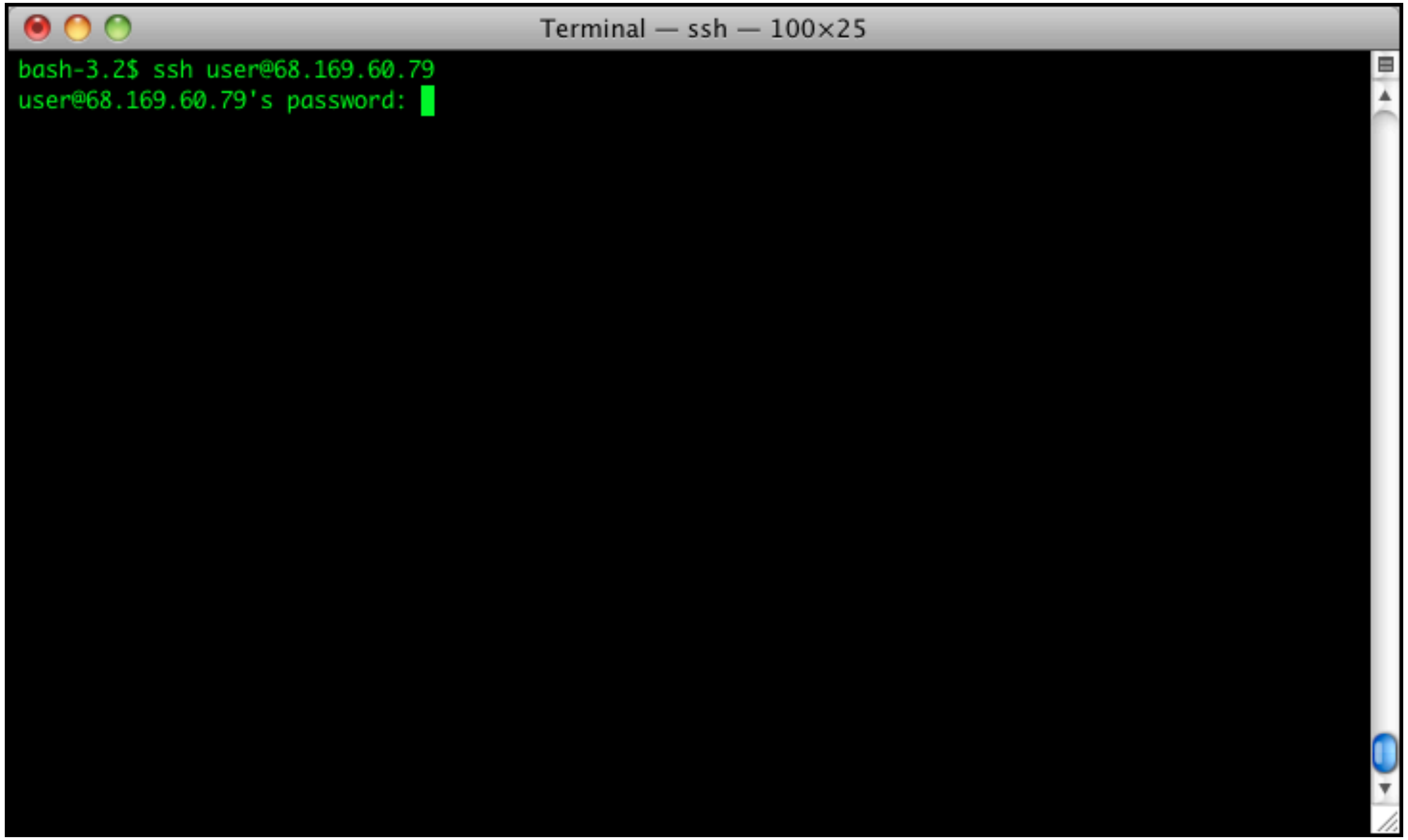
Poor utilization

Inflexible

Insecure

Not scalable

Sharing is Painful!



Sharing is Painful!

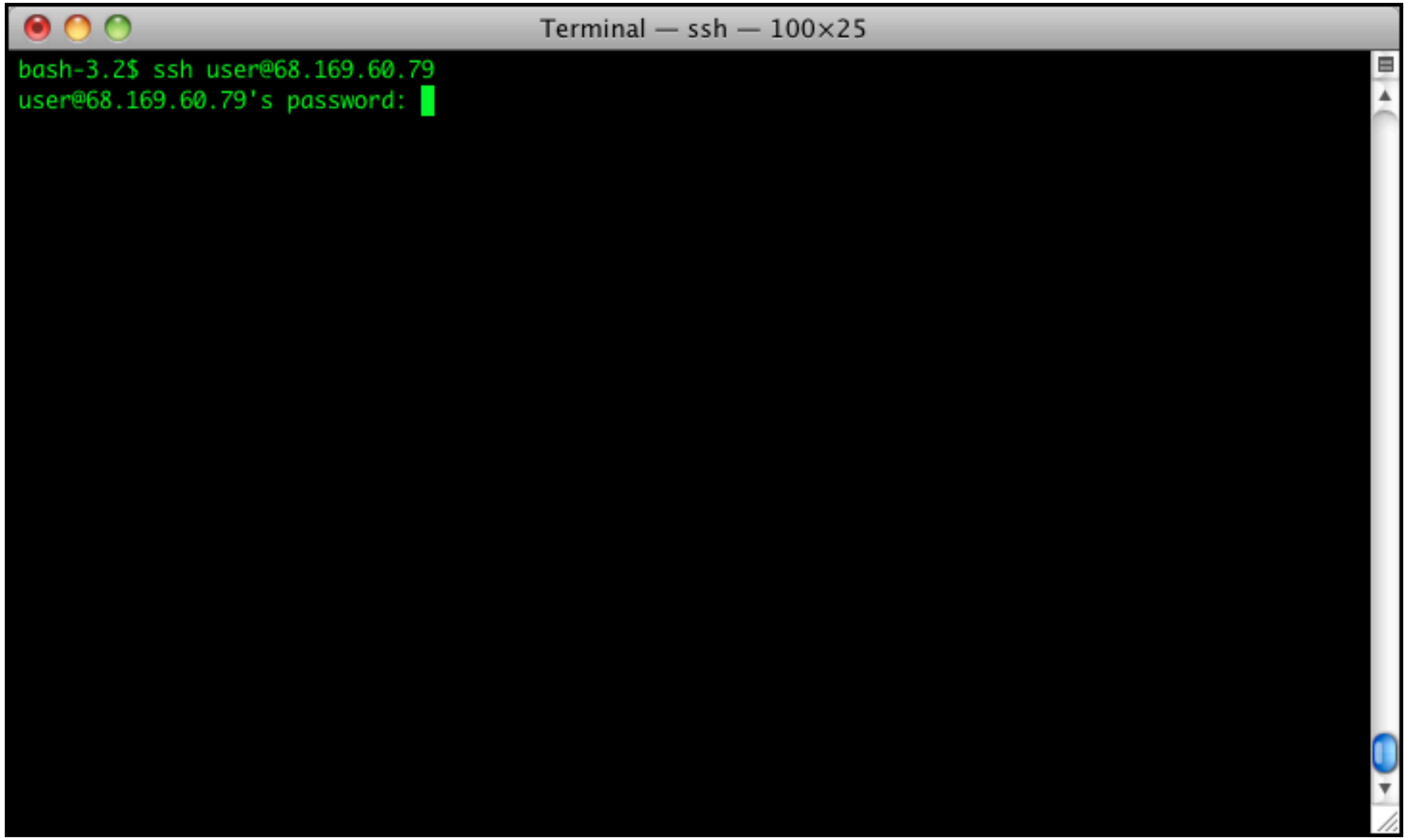
BEFORE

ssh <IP address>

Install dependencies

Manually start training script

Ad-hoc monitoring of process



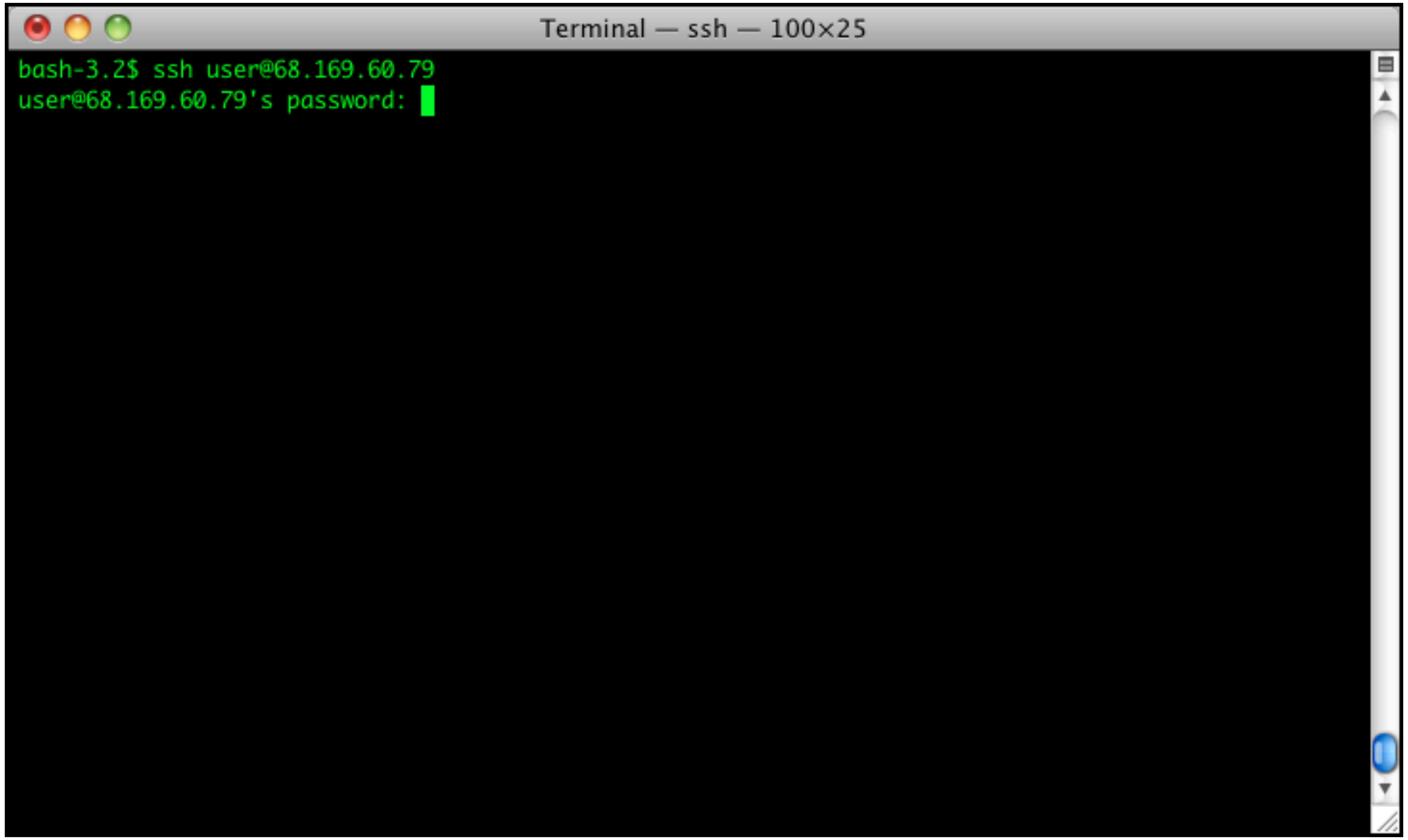
But isn't this a solved problem?

- Unified pool of GPU resources
- Run containerized workloads
- Basic task-level fault tolerance

kubernetes

MESOS

hadoop YARN


Terminal — ssh — 100×25
bash-3.2$ ssh user@68.169.60.79
user@68.169.60.79's password:

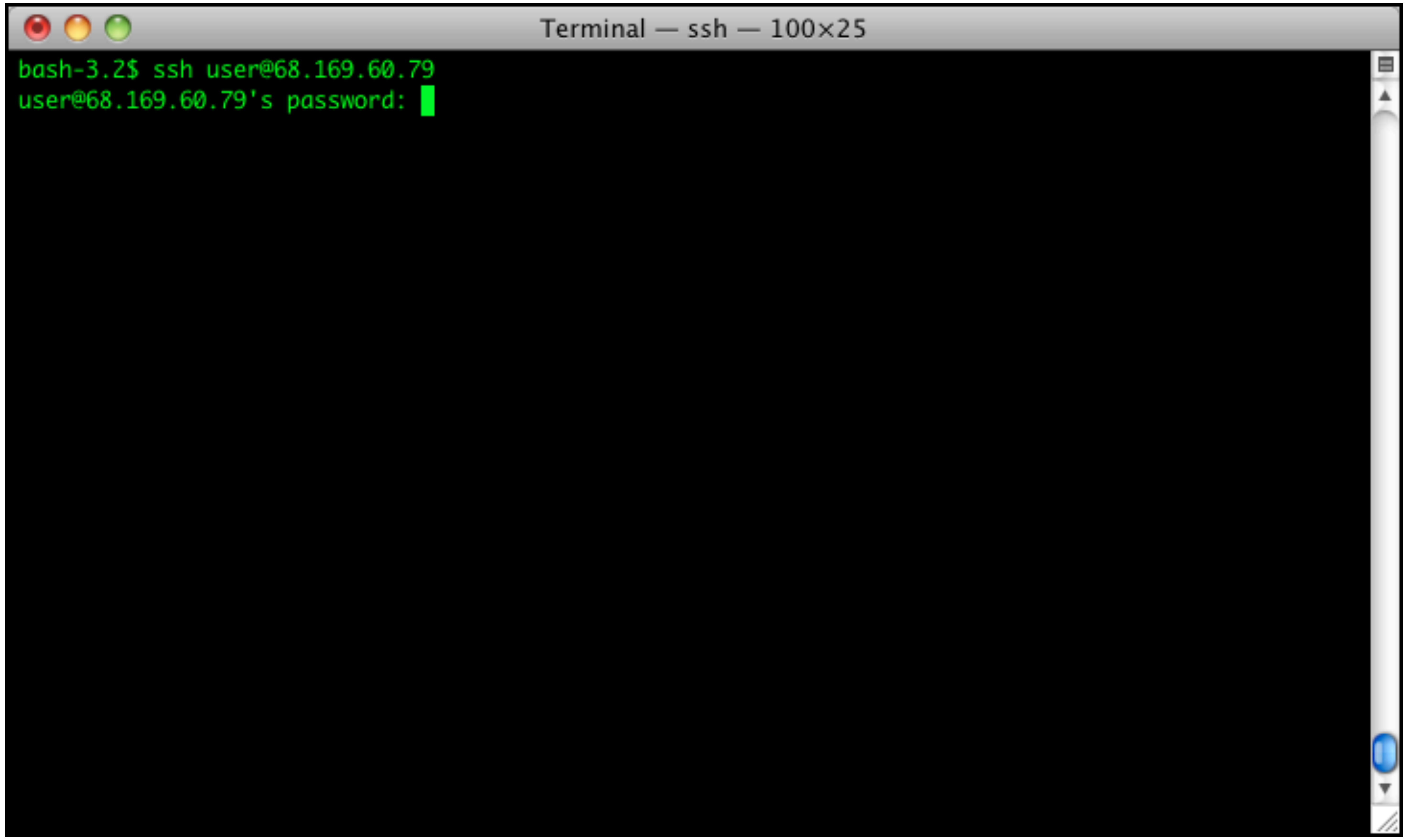
BEFORE

ssh <IP address>

Install dependencies

Manually start training script

Ad-hoc monitoring of process



docker

AFTER

Package training script and necessary dependencies into container image

Specify # GPUs needed

Send container to cluster manager, which will schedule / run it

No job migration

No auto-scaling

Not much better than a queue

The Dark Age of AI Infrastructure

Forcing users to wait for **days** to recover from faults.

Reproducing existing models is **death by a thousand cuts:** data ordering, software versions, hyperaparmeters, random seeds, model weights.

Hand-implemented, **impossibly slow** methods to find good models.

Trapping our users in systems designed to house **one user** with **rigid infrastructure.**

What else might holistic but specialized AI infrastructure look like?

- Distributed resource management for GPUs
- Managed Experiment Tracking and Visualization
- DL-Aware Fault Tolerance
- Automated Model Search
- Low Friction Multi-GPU Training

Determined AI

Our holistic but specialized platform gives AI teams the tools they need to streamline their workflows

Determined AI

Our holistic but specialized platform gives AI teams the tools they need to streamline their workflows

Best-in-class AutoML capabilities

Determined AI

Our holistic but specialized platform gives AI teams the tools they need to streamline their workflows

Best-in-class AutoML capabilities

Automated GPU resource optimization

Determined AI

Our holistic but specialized platform gives AI teams the tools they need to streamline their workflows

Best-in-class AutoML capabilities

Automated GPU resource optimization

Seamless reproducibility and collaboration

Determined AI

Our holistic but specialized platform gives AI teams the tools they need to streamline their workflows

Best-in-class AutoML capabilities

Automated GPU resource optimization

Seamless reproducibility and collaboration

Support for cloud, on-premise, hybrid usage

Determined AI

Our holistic but specialized platform gives AI teams the tools they need to streamline their workflows

Best-in-class AutoML capabilities

Automated GPU resource optimization

Seamless reproducibility and collaboration

Support for cloud, on-premise, hybrid usage

Support of TensorFlow, PyTorch Keras

Determined AI

Thank you. Come talk to us!

Learn more at https://determined.ai/