

Keras: the next five years

ScaledML - February 27th, 2020

@fchollet

TensorFlow:
a powerful foundation



Tensor manipulation

```
with tf.GradientTape() as outer_tape:
  with tf.GradientTape() as tape:
    c = tf.sqrt(tf.square(a) + tf.square(b))
    dc_da = tape.gradient(c, a)
  d2c_da2 = outer_tape.gradient(dc_da, a)
```



Performance & scale

Deploy ML anywhere

Servers



TensorFlow
Extended

Edge devices



TensorFlow
Lite

JavaScript



TensorFlow
.JS

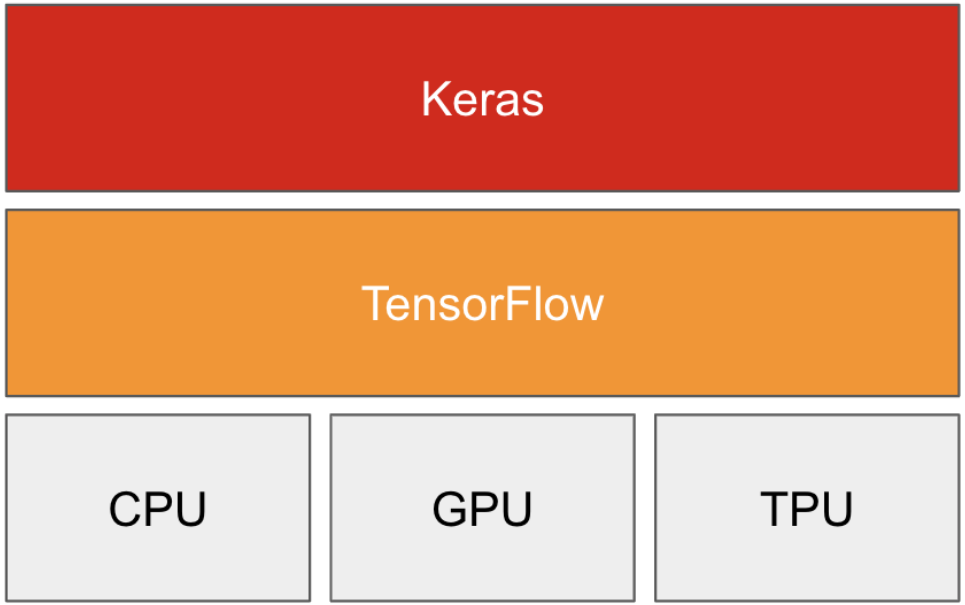
A **tensor-manipulation** framework (like NumPy)

With **automatic differentiation**

Keras:
radically productive deep learning



The relationship between Keras and TensorFlow



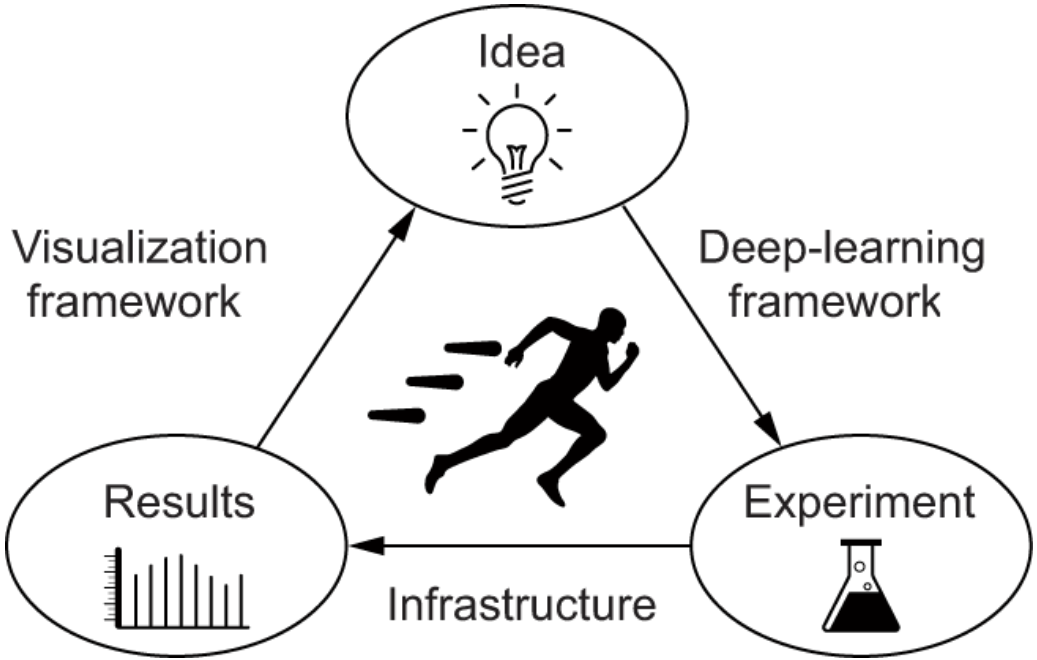
Deep learning development: layers, models, optimizers, losses, metrics...

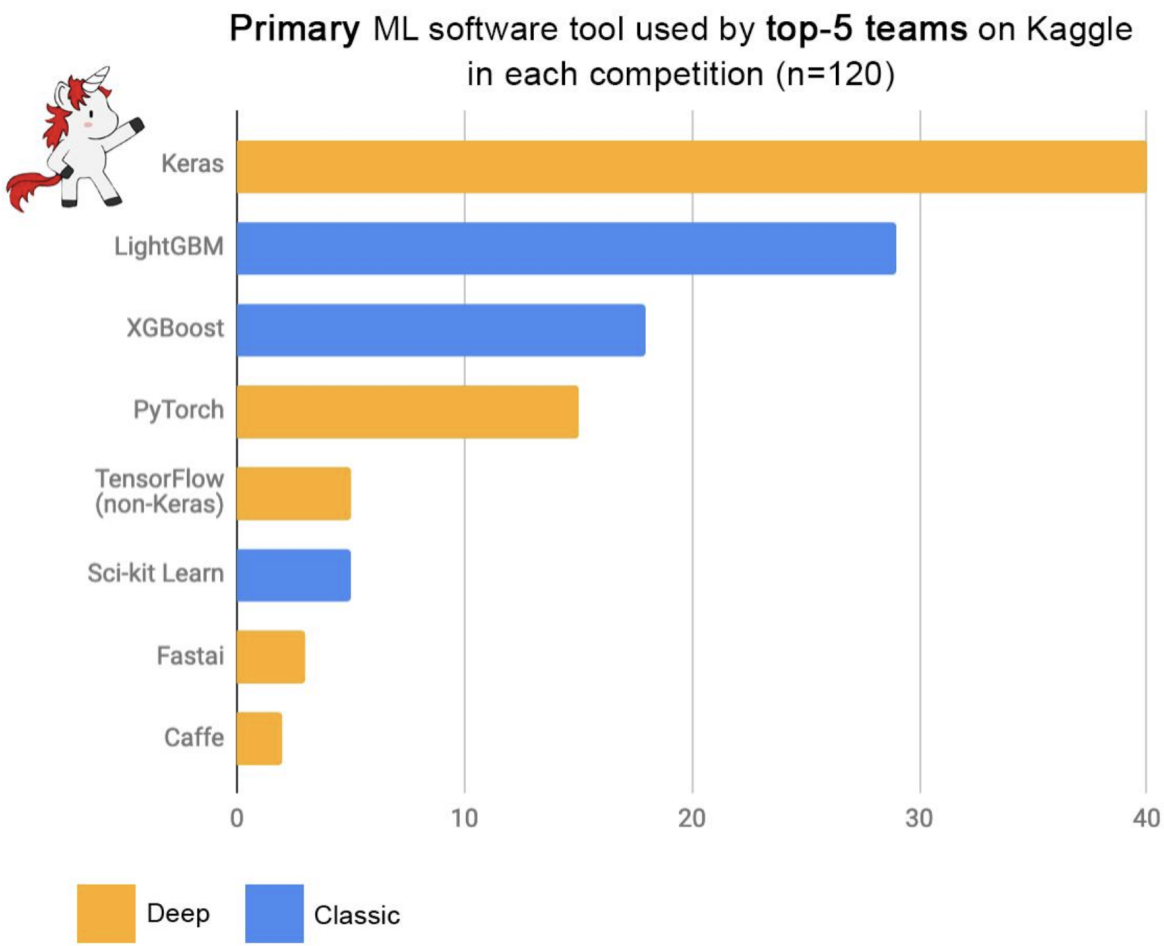
Tensor manipulation infrastructure: tensors, variables, automatic differentiation, distribution...

Hardware: execution

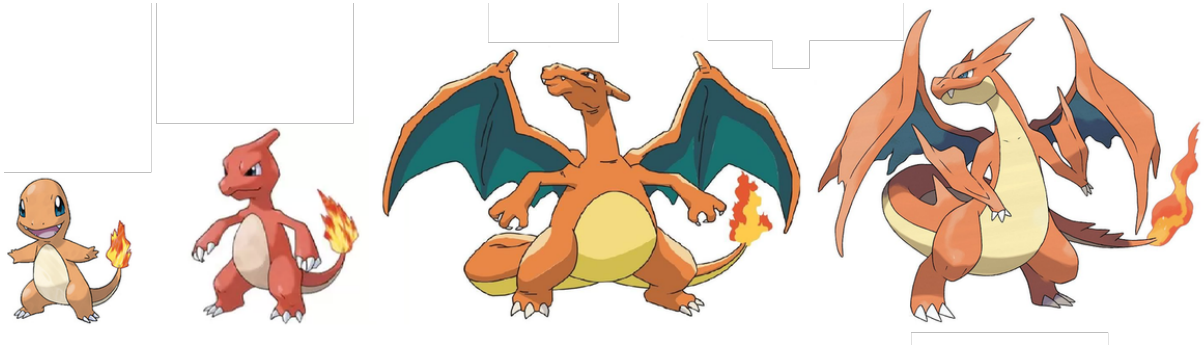
Productive high-level API

= build great products,
win competitions, get papers published

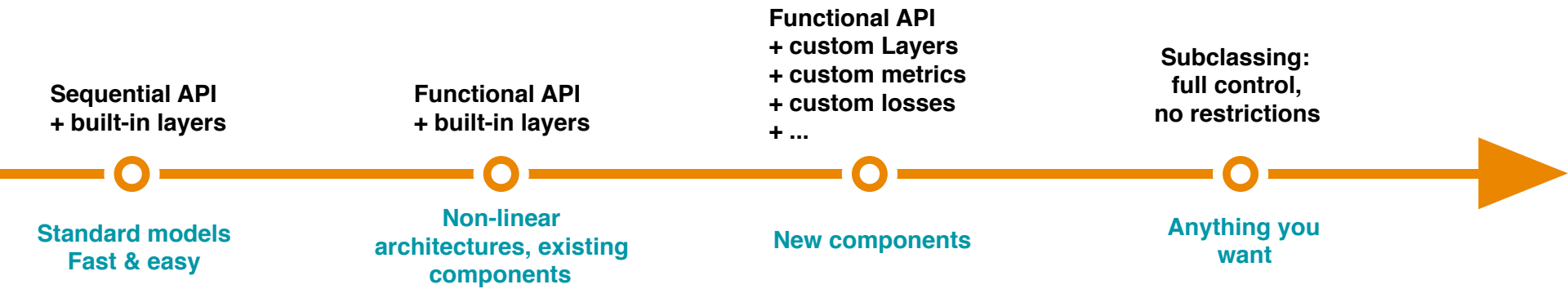



Primary ML software tool used by top-5 teams on Kaggle in each competition (n=120)
Keras
LightGBM
XGBoost
PyTorch
TensorFlow (non-Keras)
Sci-kit Learn
Fastai
Caffe
0
10
20
30
40
Deep
Classic

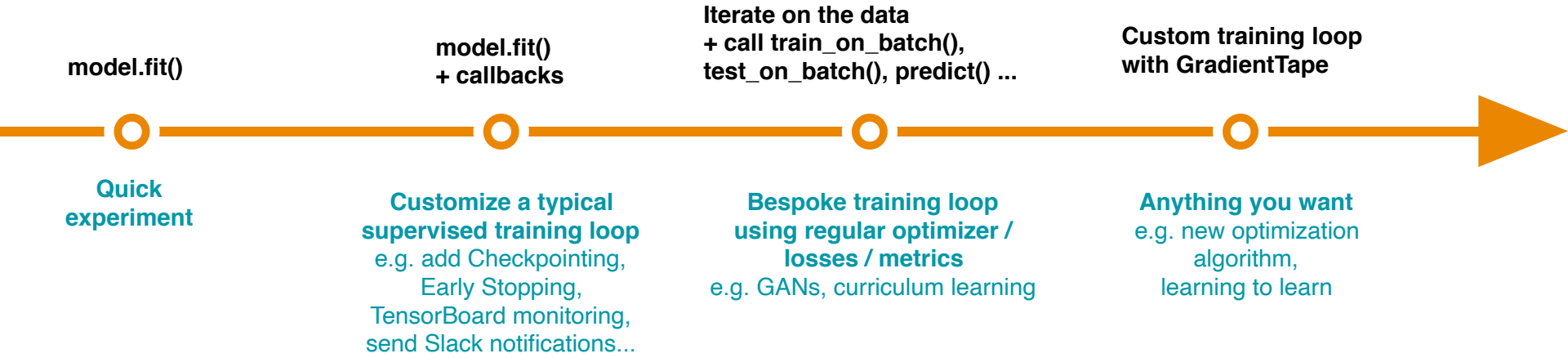
The Keras API philosophy:



Model building: from **simple** to **arbitrarily flexible**



Model training: from **simple** to **arbitrarily flexible**



Researchers
Data scientists
Engineers

The next five years

Where's deep learning headed?

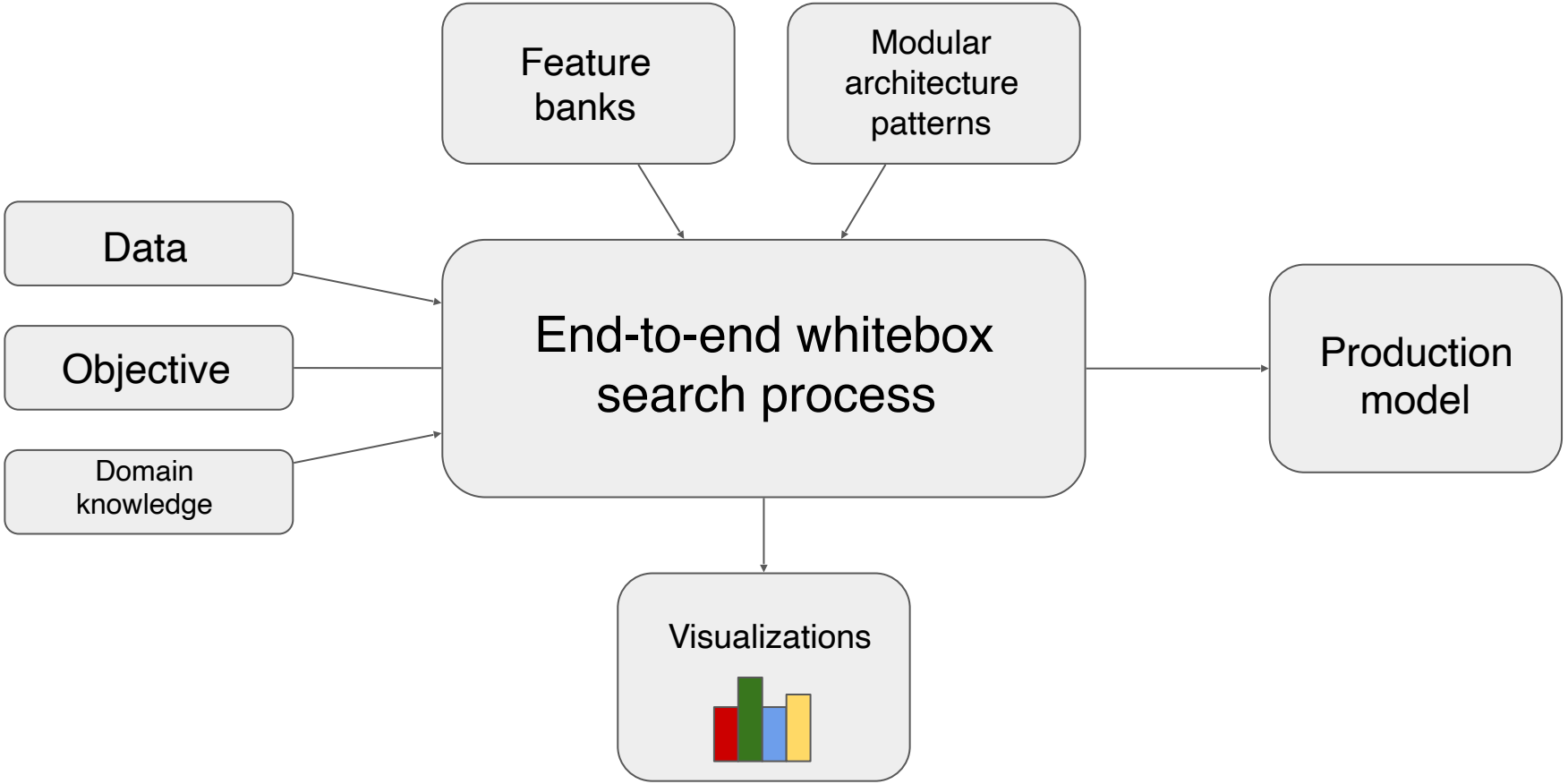
Leveraging current tech to solve every problem it can solve

#1 Automation
#2 Scale & Cloud

#1: Automation

- Hyperparameter tuning
- Architecture search
- Feature banks & lifelong learning
- AutoML

Towards ever **higher-level** workflows


Feature banks
Modular architecture patterns
Data
Objective
Domain knowledge
End-to-end whitebox search process
Production model
Visualizations

Keras Tuner

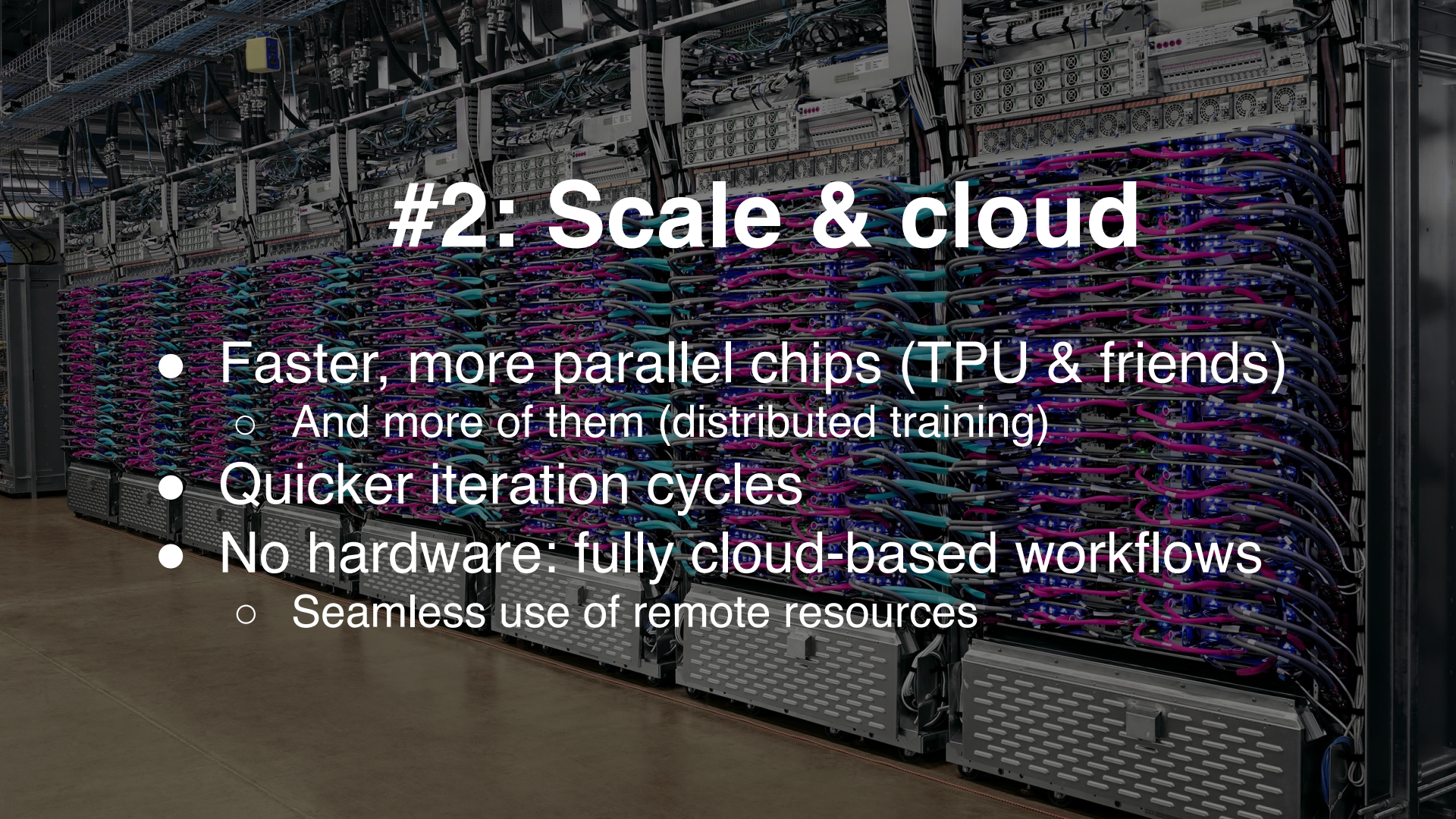
- Define-by-run dynamic search spaces
- Built-in search strategies: Hyperband, bayesian, random search
- Define your own strategies!
- Large-scale distributed search
- Built-in "tunable models" (application templates: "just add data")



AutoKeras

```
# Prepare the data.
(x_train, y_train), (x_test, y_test) = imdb_raw()
print(x_train.shape)  # (25000,)
print(y_train.shape)  # (25000, 1)
print(x_train[0][:50])  # <START> this film was just brilliant casting <UNK>

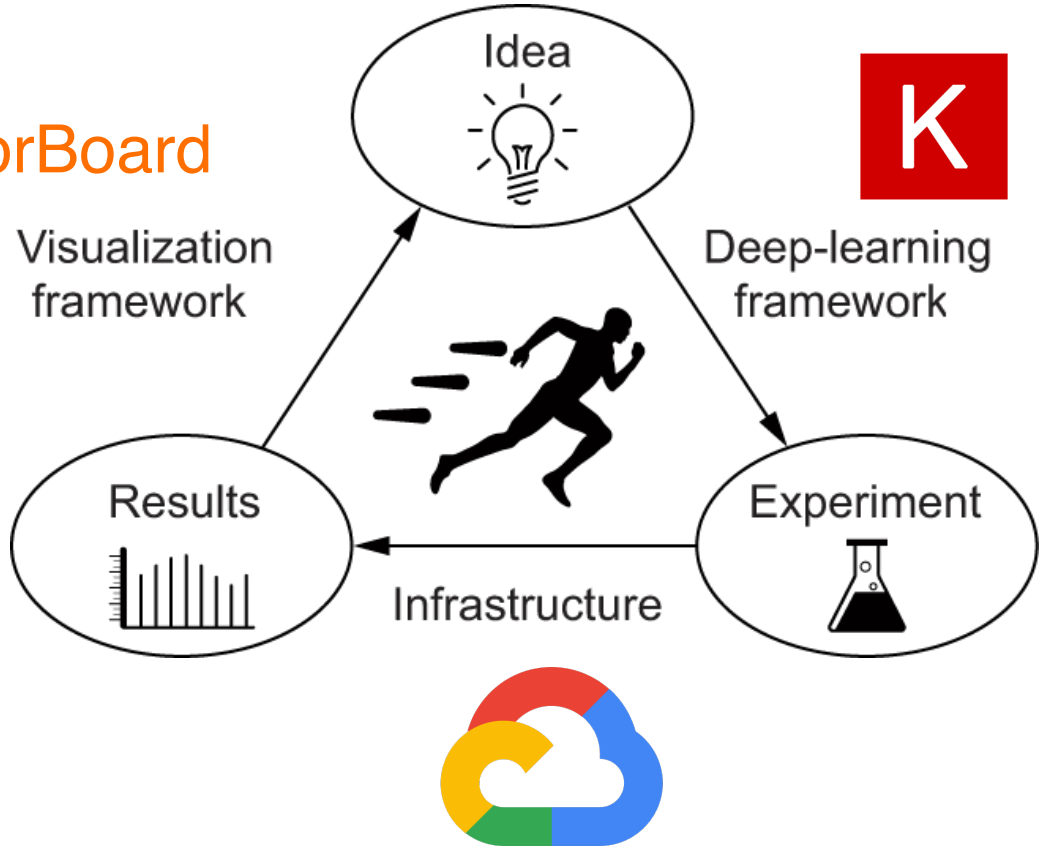
# Initialize the TextClassifier
clf = ak.TextClassifier(max_trials=3)
# Search for the best model.
clf.fit(x_train, y_train, epochs=2)
# Evaluate on the testing data.
print('Accuracy: {accuracy}'.format(accuracy=clf.evaluate(x_test, y_test)))
```



#2: Scale & cloud

- Faster, more parallel chips (TPU & friends)
  - And more of them (distributed training)
- Quicker iteration cycles
- No hardware: fully cloud-based workflows
  - Seamless use of remote resources

TensorBoard



Keras

Cloud Keras

`tf.keras.cloud.run(**config)`

- From inside a notebook (including Colab)
- From inside a local script
- From a shell, targeting a local script or notebook file

```
tf.keras.cloud.run(
    chief_config=MachineConfig(
            cpu_cores=4, memory=15),
    worker_config=MachineConfig(
            cpu_cores=8,
            memory=30,

accelerator_type=NVIDIA_TESLA_P100,
            accelerator_count=4)
    worker_count=8,
    output_dir=my_remote_dir,
    stream_logs=True)
```

Cloud Keras for training & hyperparameter tuning

- Instantly train on a beefy machine on GCP (e.g. 8 P100s)
- Or on a multi-worker GPU cluster
- Or on a TPU pod

Works for `model.fit()`, custom training loops, and **Keras Tuner tuning runs**



#3: Into the real world

- Ever wider range of applications
- Mobile, browser, embedded devices
- Resource-efficient models
- Personalization: on-device training & privacy
- Federated learning

Portable models with **Keras Preprocessing Layers**

- Text vectorization
  - String standardization / cleaning
  - Tokenization
  - Indexing
  - ngrams, TF-IDF
- Image preprocessing and augmentation
- Structured data preprocessing
  - Indexing
  - Hashing
  - Discretization

end-to-end models

TF.js, TF Lite

Edge devices



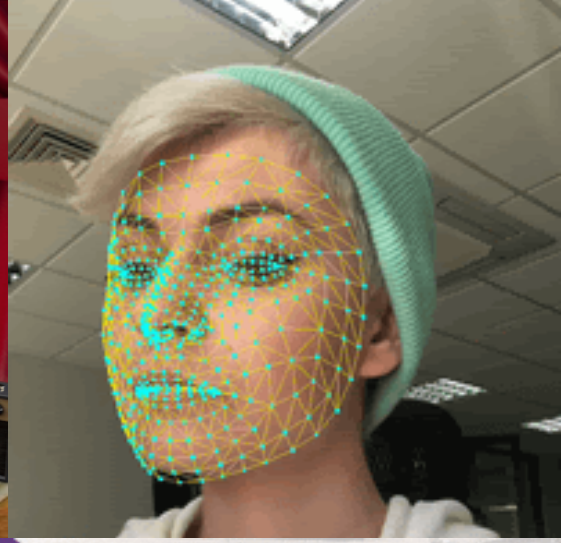
TensorFlow
Lite

JavaScript



TensorFlow
.JS


Harmonize





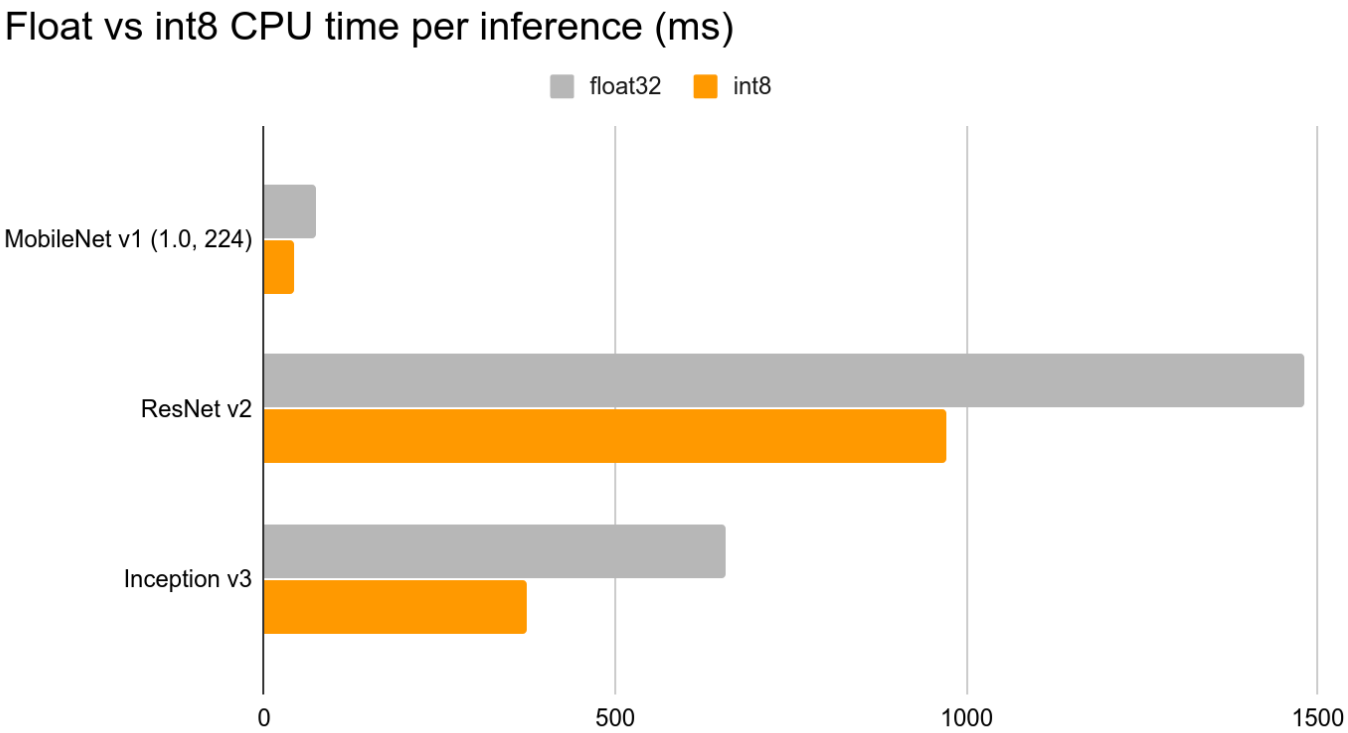


Model optimization toolkit

- Post-training quantization
- Pruning-aware training



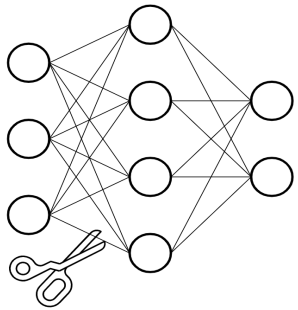
Post-training quantization



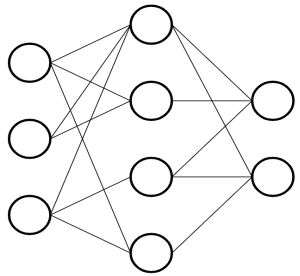
Pruning-aware training

```
model = build_your_model()
pruning_schedule = tfmot.sparsity.keras.PolynomialDecay(
      initial_sparsity=0.0, final_sparsity=0.5,
      begin_step=2000, end_step=4000)

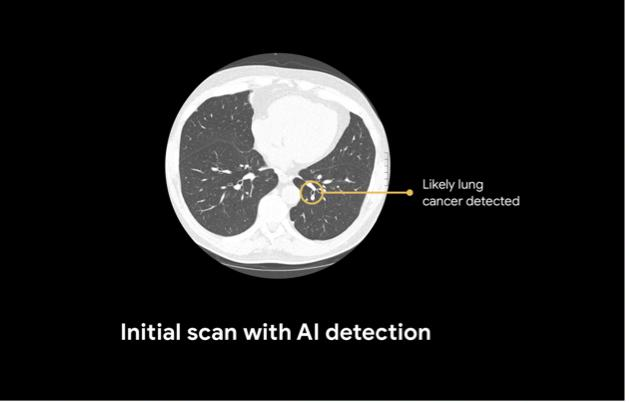
model_for_pruning = tfmot.sparsity.keras.prune_low_magnitude(
      model, pruning_schedule=pruning_schedule)
...
model_for_pruning.fit(...)
```



Before pruning



After pruning


Likely lung
cancer detected
Initial scan with AI detection

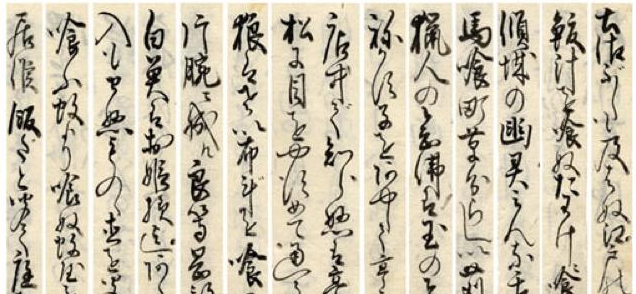










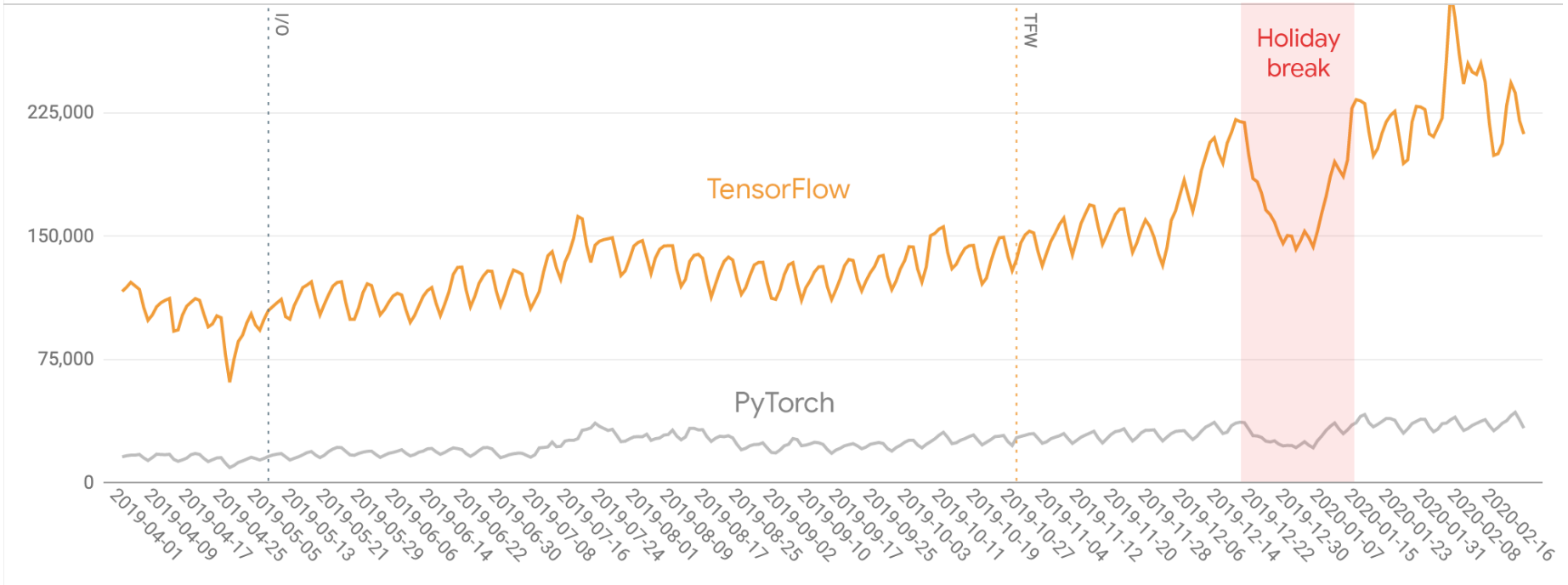




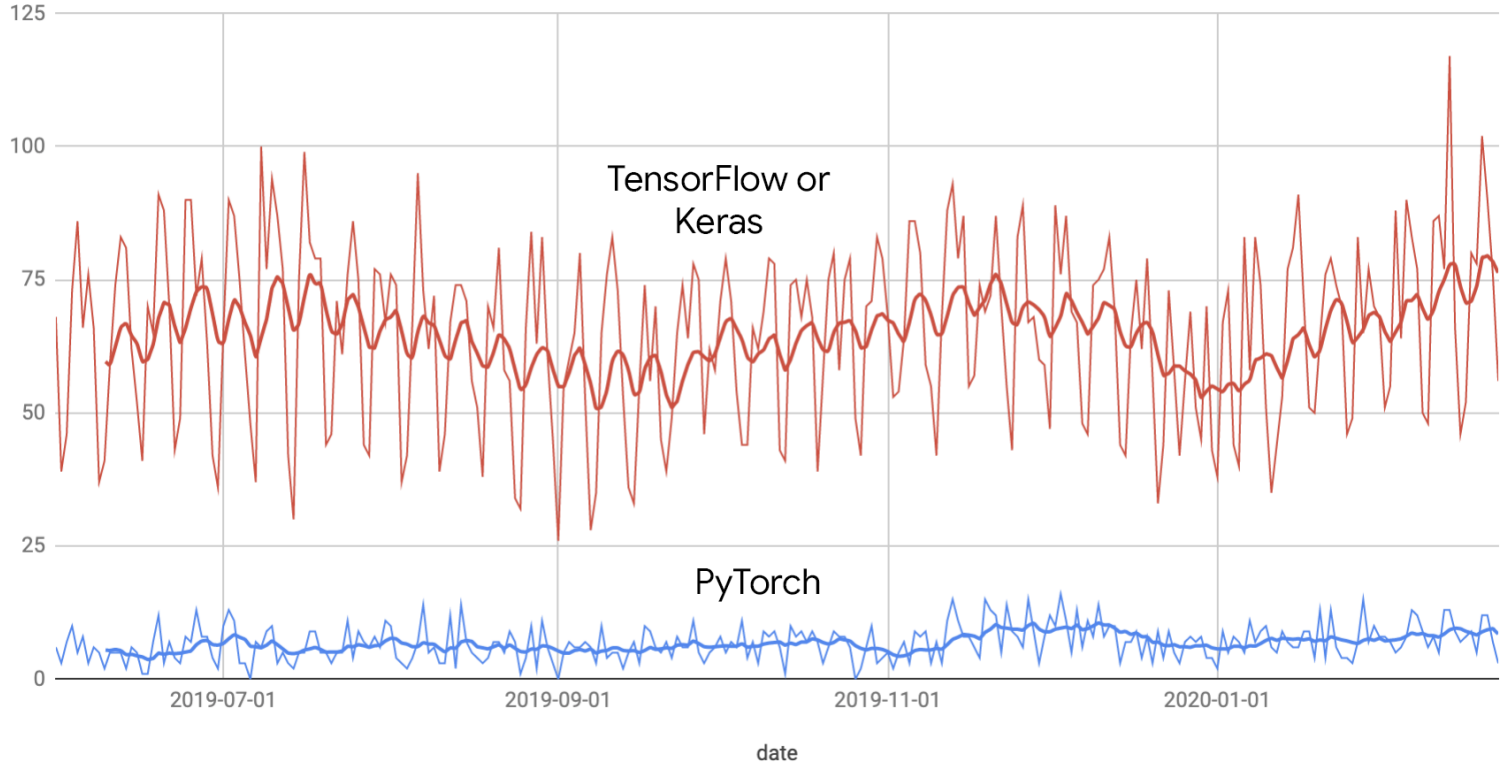
Thank you!

Bonus: adoption stats!

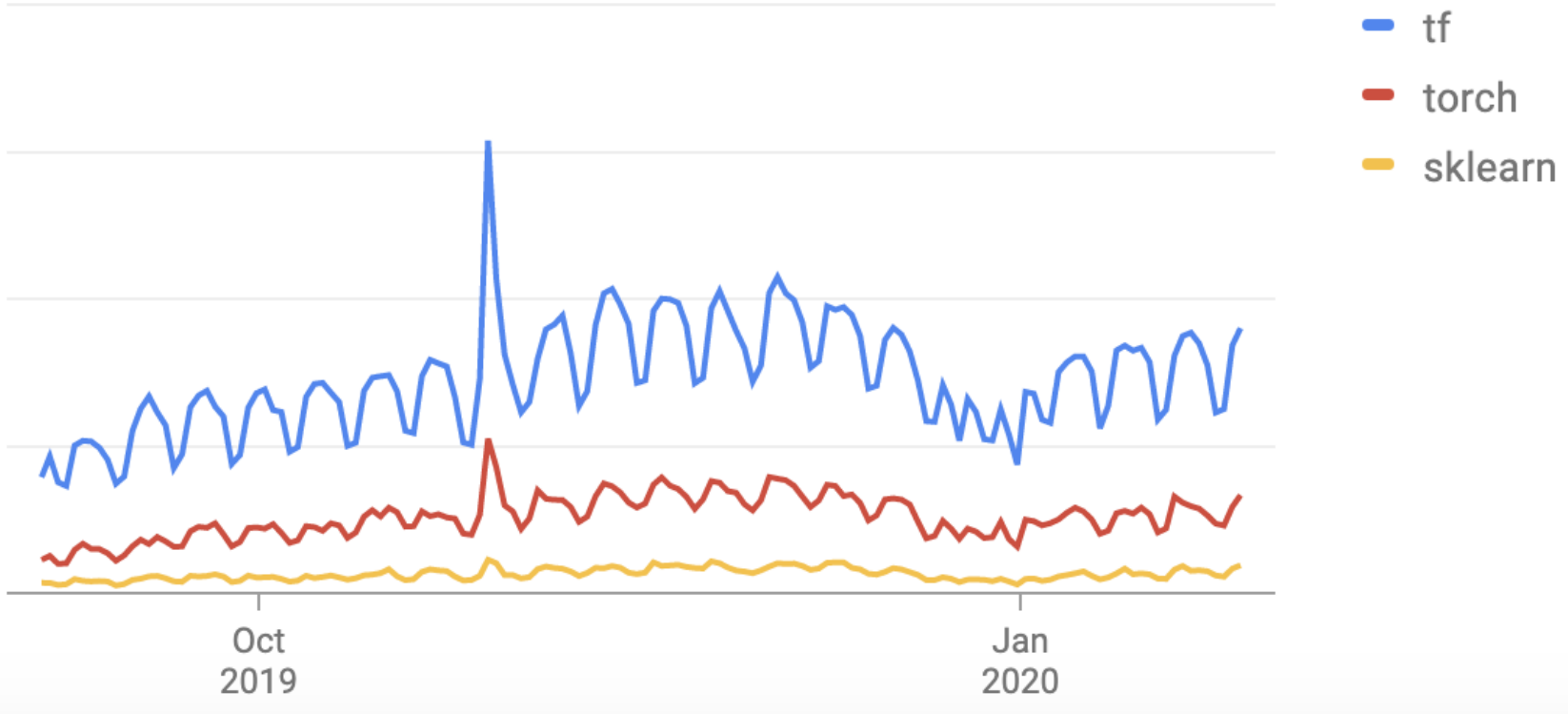
PyPI Downloads (per day, EMA)



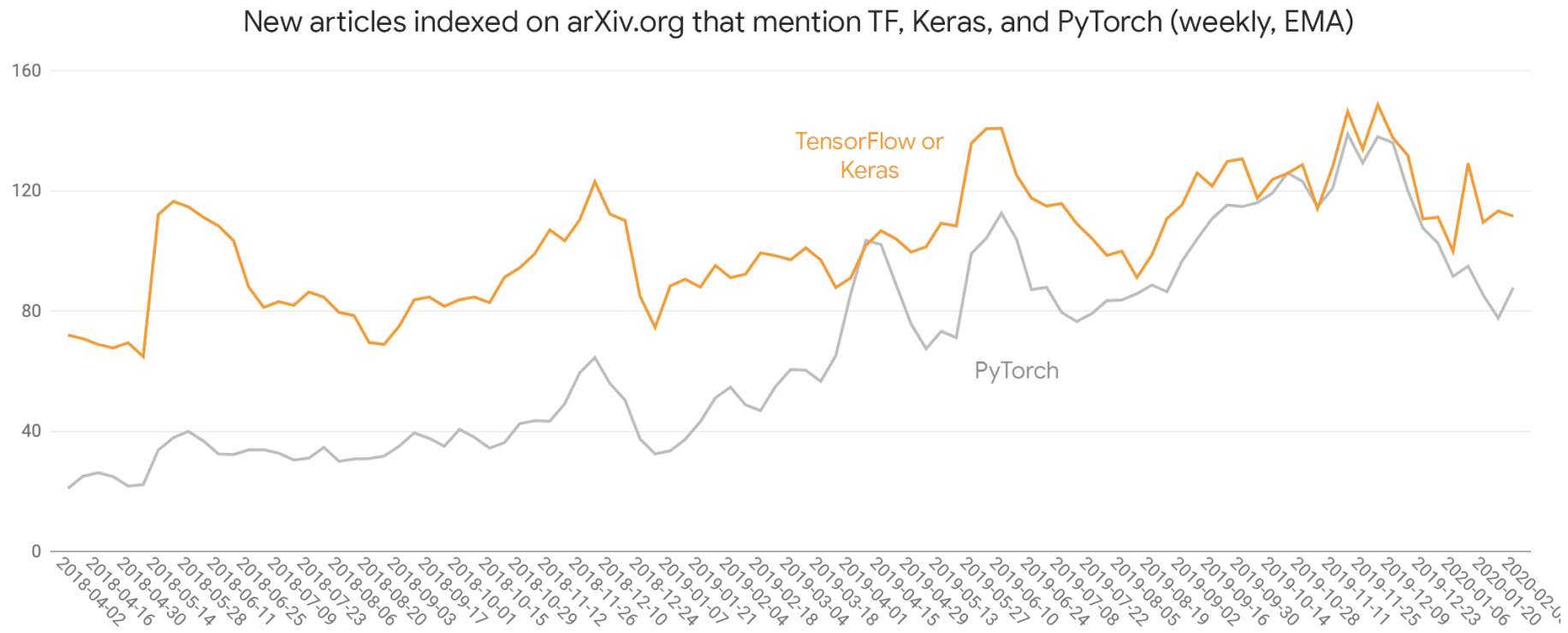
StackOverflow (new questions per day)



Colab usage (~mostly research, students)



ArXiv (~pure ML research)



Google Scholar (~applied ML research)

