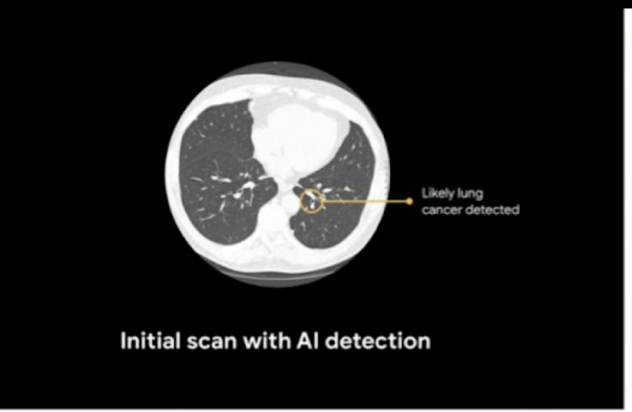
Keras & TensorFlow

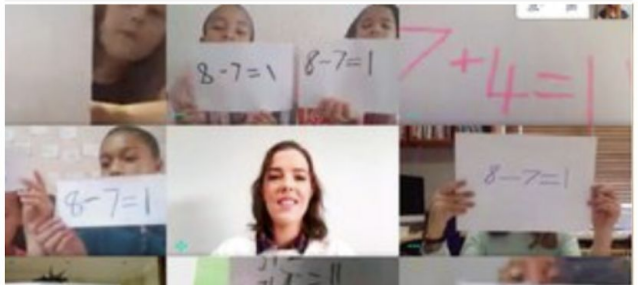
The next five years

Francois Chollet

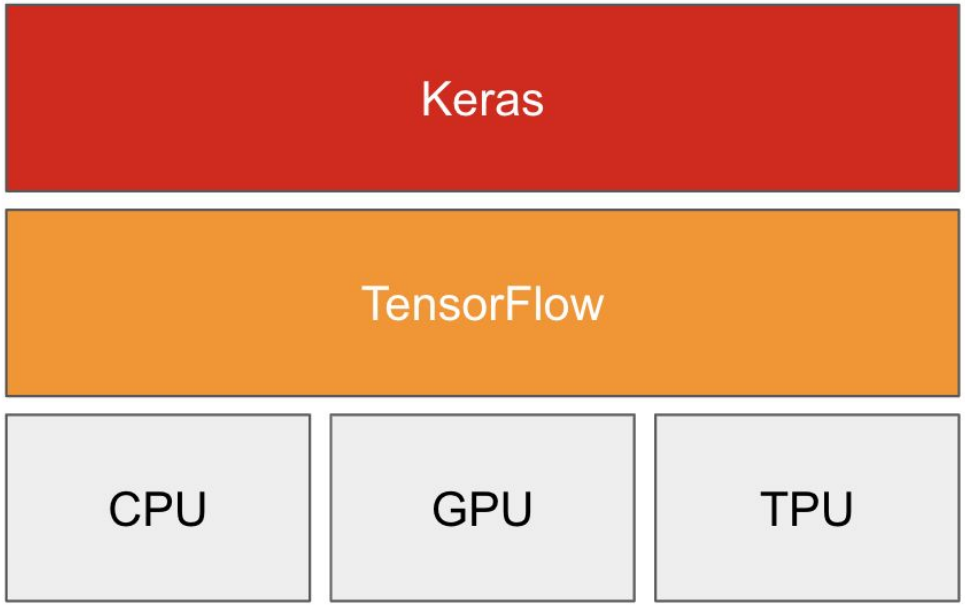
GTC 2021


Likely lung
cancer detected
Initial scan with AI detection

TensorFlow


8-7=1
8-7=1
7+4=1
8-7=1
8-7=1

The relationship between Keras and TensorFlow



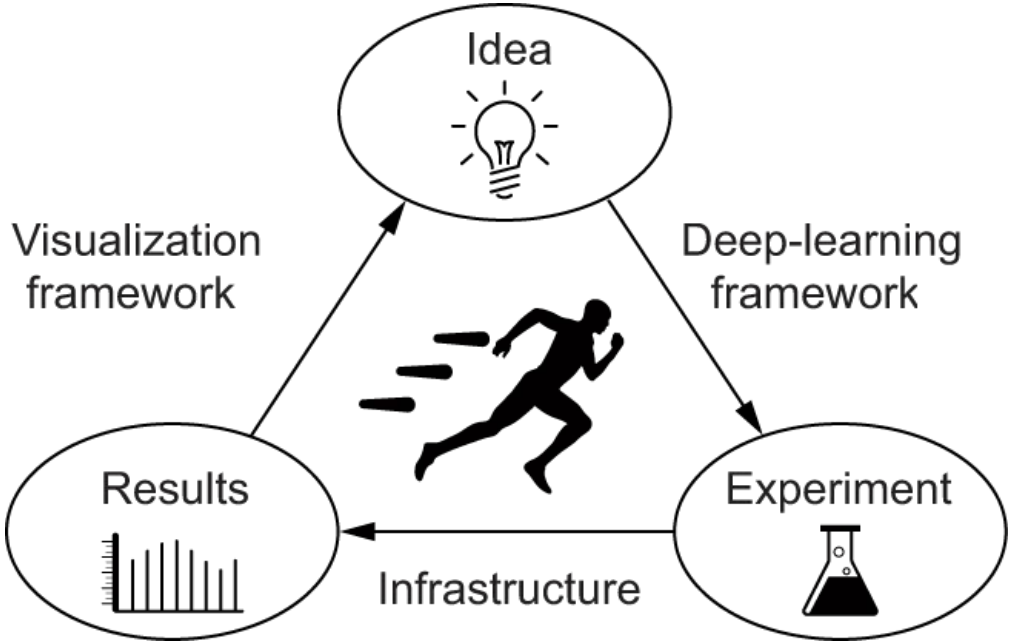
Deep learning development: layers, models, optimizers, losses, metrics...

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

Hardware: execution

The loop of progress

build great products,
win competitions, get papers published



Who uses Keras?

3 groups of users

Basic users

- As little code as possible
- Batteries included
- Best-practices baked-in by default

Engineers

- Flexible model architecture
- Flexible training logic
- Scale and speed
- Deployment-readiness

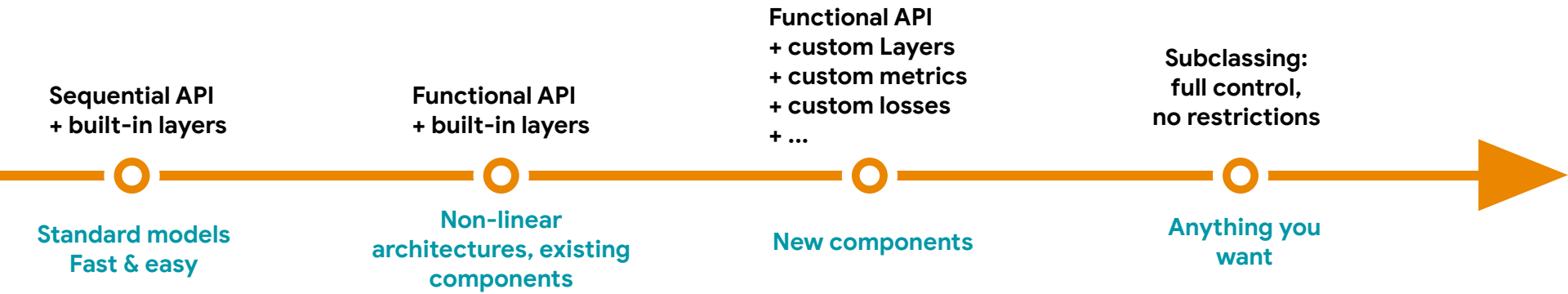
Researchers

- Don't want a framework, just a toolbox
- Low-level control of every detail
- Simple, consistent programming mental models

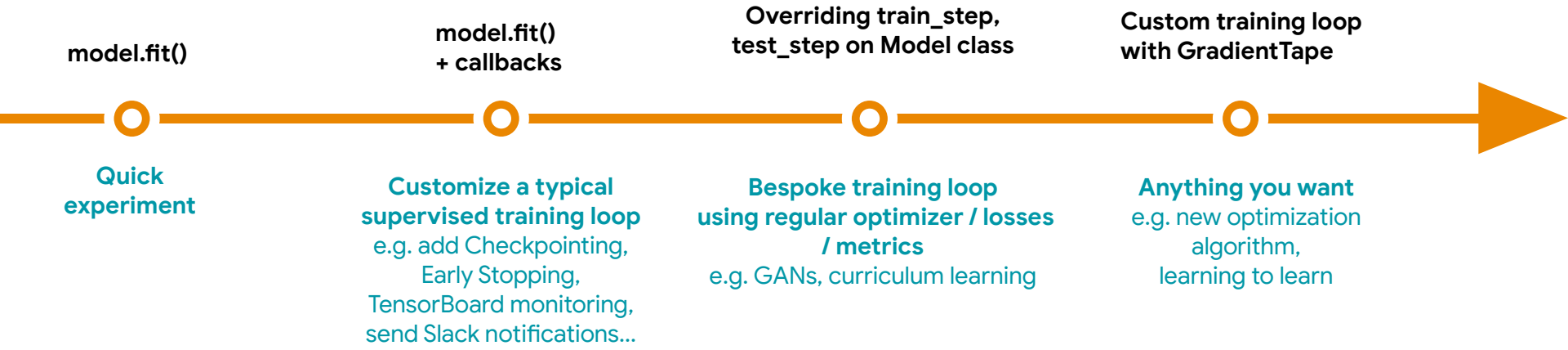
The Keras API philosophy:

Progressive disclosure of complexity

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



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




Researchers
Data scientists
Engineers

The next five years

Where is Deep Learning headed?

Leveraging current tech to solve **every problem** it can solve

Ecosystems of reusable parts

Increasing automation

Larger-scale workflows (in the cloud)

Real-world deployment

An ever-growing **ecosystem** of **reusable** parts

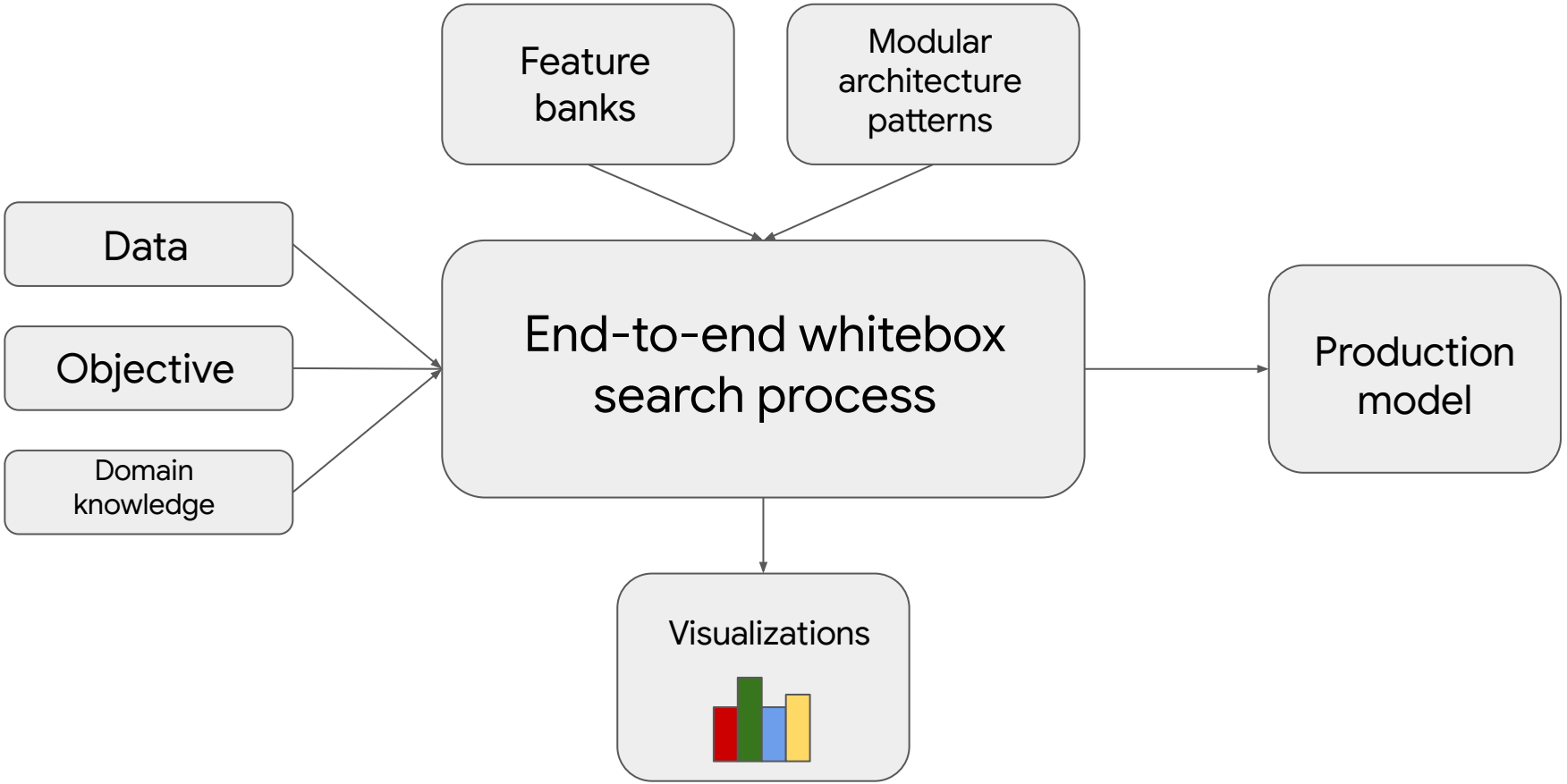
- Reusable domain-specific functionality
  - **KerasCV**, **KerasNLP**
- Larger banks of reusable pre-trained models
  - Expanded **Keras Applications**
- Feature banks

Train once, reuse forever

Increasing **automation**

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

*Towards ever **higher-level** workflows*

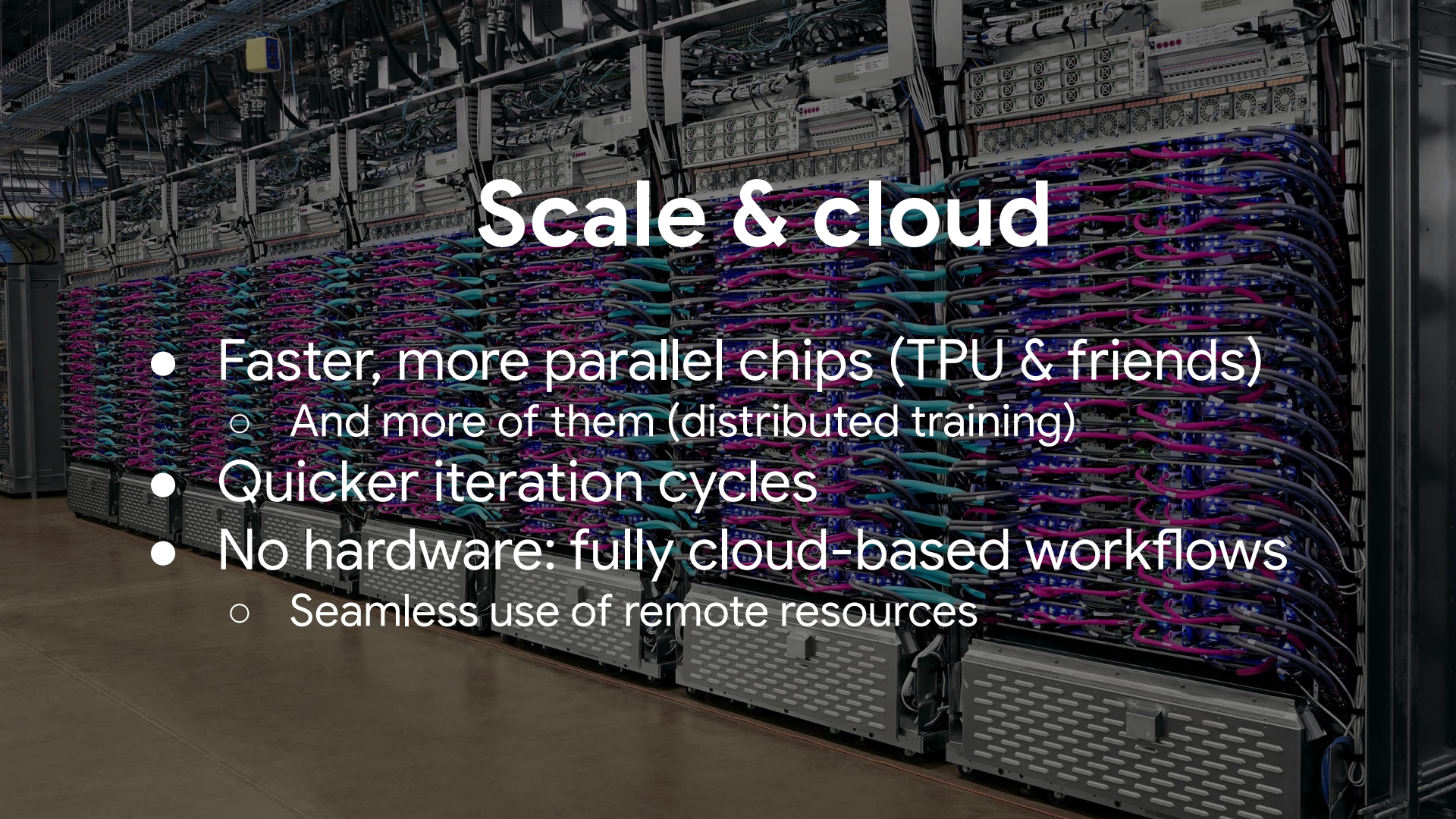

Feature banks
Modular architecture patterns
Data
Objective
Domain knowledge
End-to-end whitebox search process
Production model
Visualizations

KerasTuner

- 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

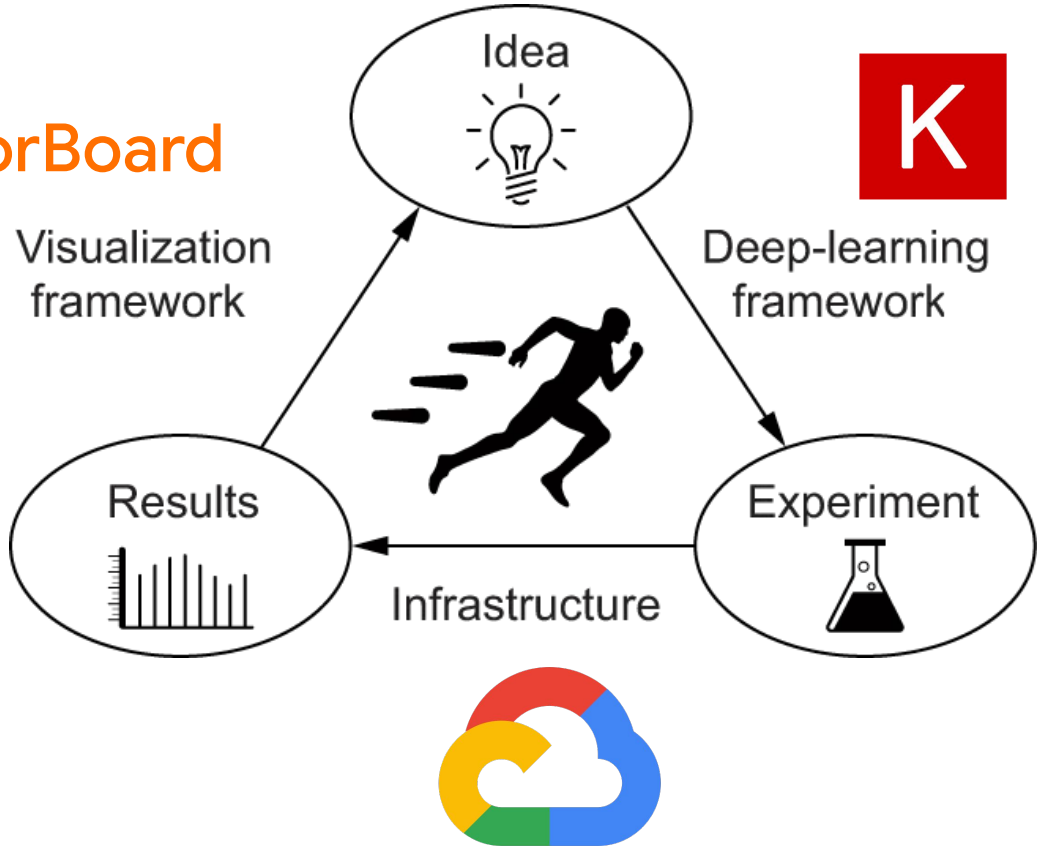
```
# Prepare the data.
(x_train, y_train), (x_test, y_test) = imdb_raw()
print(x_train.shape)  # (25000,)
print(y_train.shape)  # (25000, 1)
print(x_train[0][:50])  # <START> this film was just brilliant casting <UNK>

# 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)))
```



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

TensorFlow Cloud

`tfc.run(**config)`

- From inside a notebook
  - E.g. **Colab** notebooks
  - **Kaggle** notebooks
- From inside a local script
- From a shell, targeting a local script or notebook file

```
tfc.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)
```

TF Cloud 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 **KerasTuner tuning runs**

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

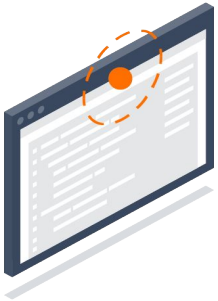
Reduced training/serving skew

TF.js, TF Lite

Edge devices

TensorFlow
Lite

JavaScript



TensorFlow
.JS

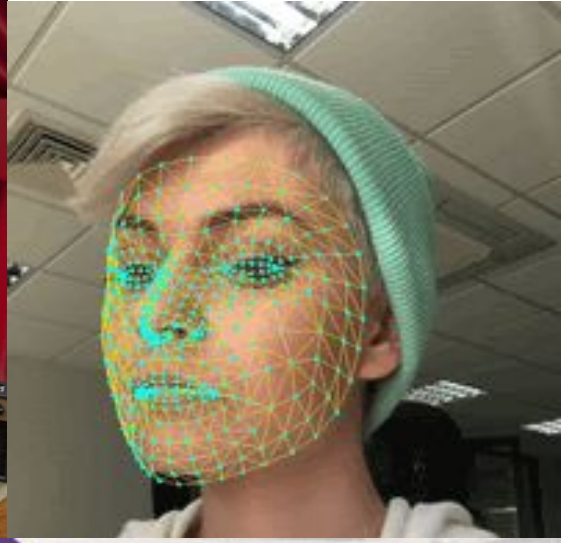
Latest on-device ML with TensorFlow Lite

Hold For Me on new Pixel 5/4a

OCR translation in Google Translate

HDR+ on Pixel 4a

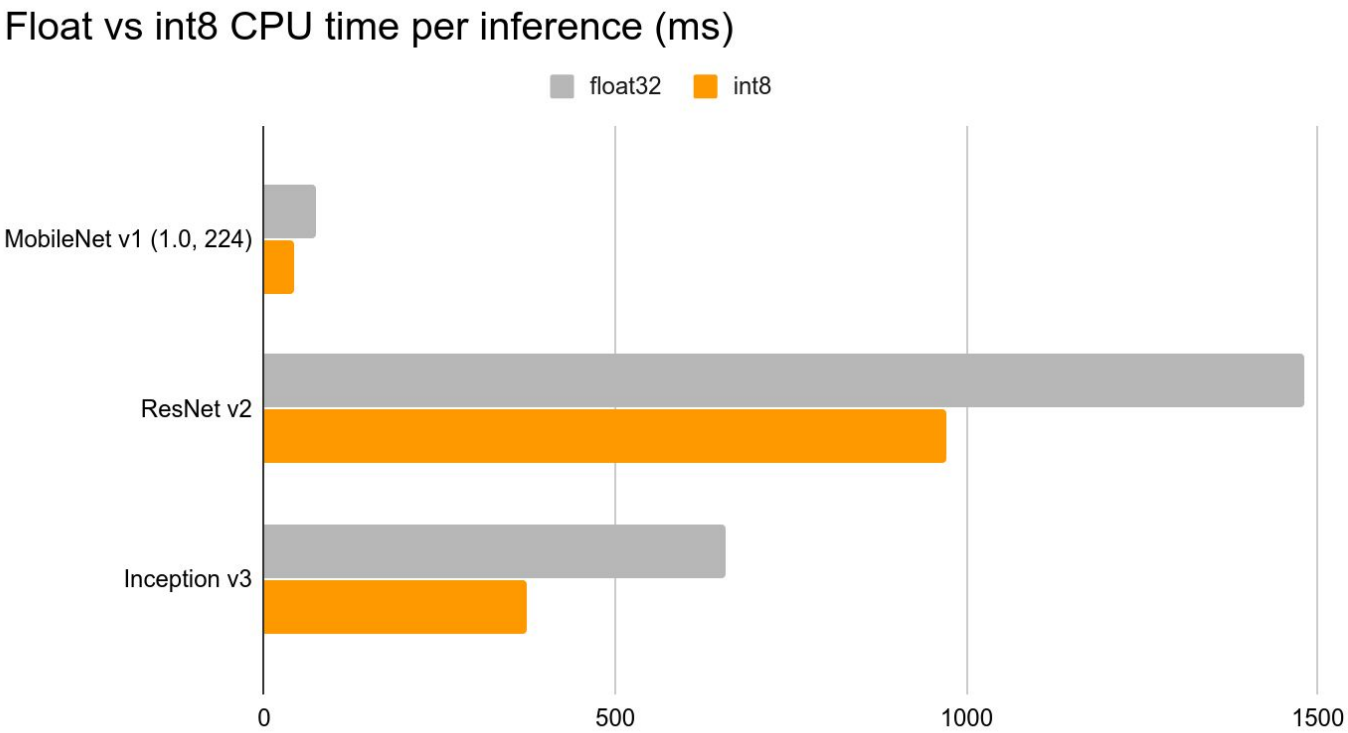

Harmonize



Model optimization toolkit

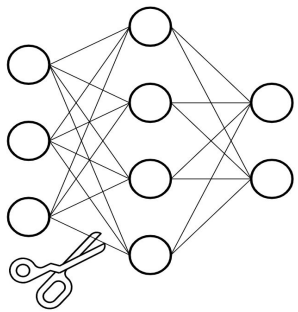
- Post-training quantization
- Pruning-aware training

Post-training quantization

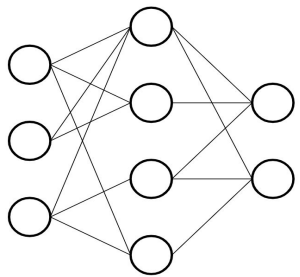


Pruning-aware training

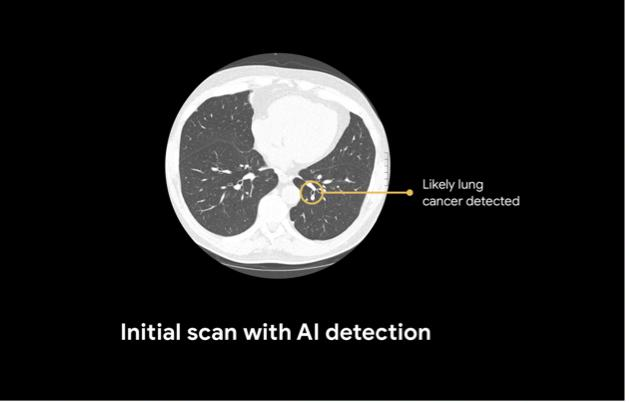
```
model = build_your_model()
pruning_schedule = tfmot.sparsity.keras.PolynomialDecay(
      initial_sparsity=0.0, final_sparsity=0.5,
      begin_step=2000, end_step=4000)

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

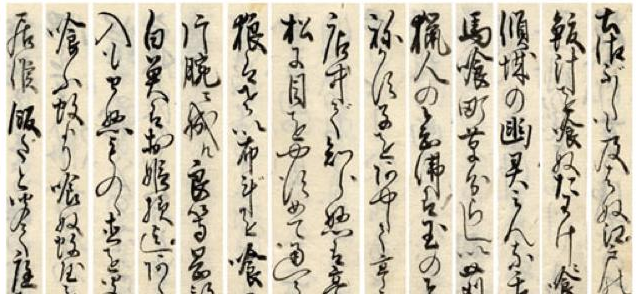


Before pruning



After pruning


Likely lung
cancer detected
Initial scan with AI detection



Thank you!