텐서플로우 2

케라스와 함께 새로운 시대의 출발

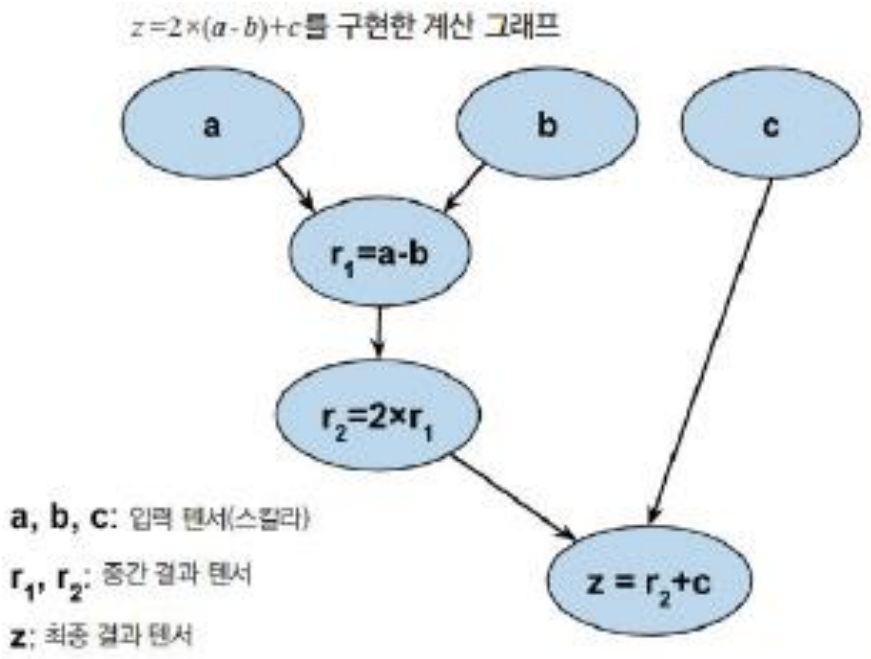
Sept 2020

텐서플로우 2.0

텐서플로우 2

- 2015년 11년에 구글에서 공개한 데이터플로우 그래프와 AutoDiff를 기반으로 하는 머신러닝 프레임워크 (Define and Run 스타일)

- C++ core와 Python API 기반이나 Keras 등 여러 고수준 API도 도입



텐서플로우 2

- 2019년 10월 텐서플로우 2.0 발표

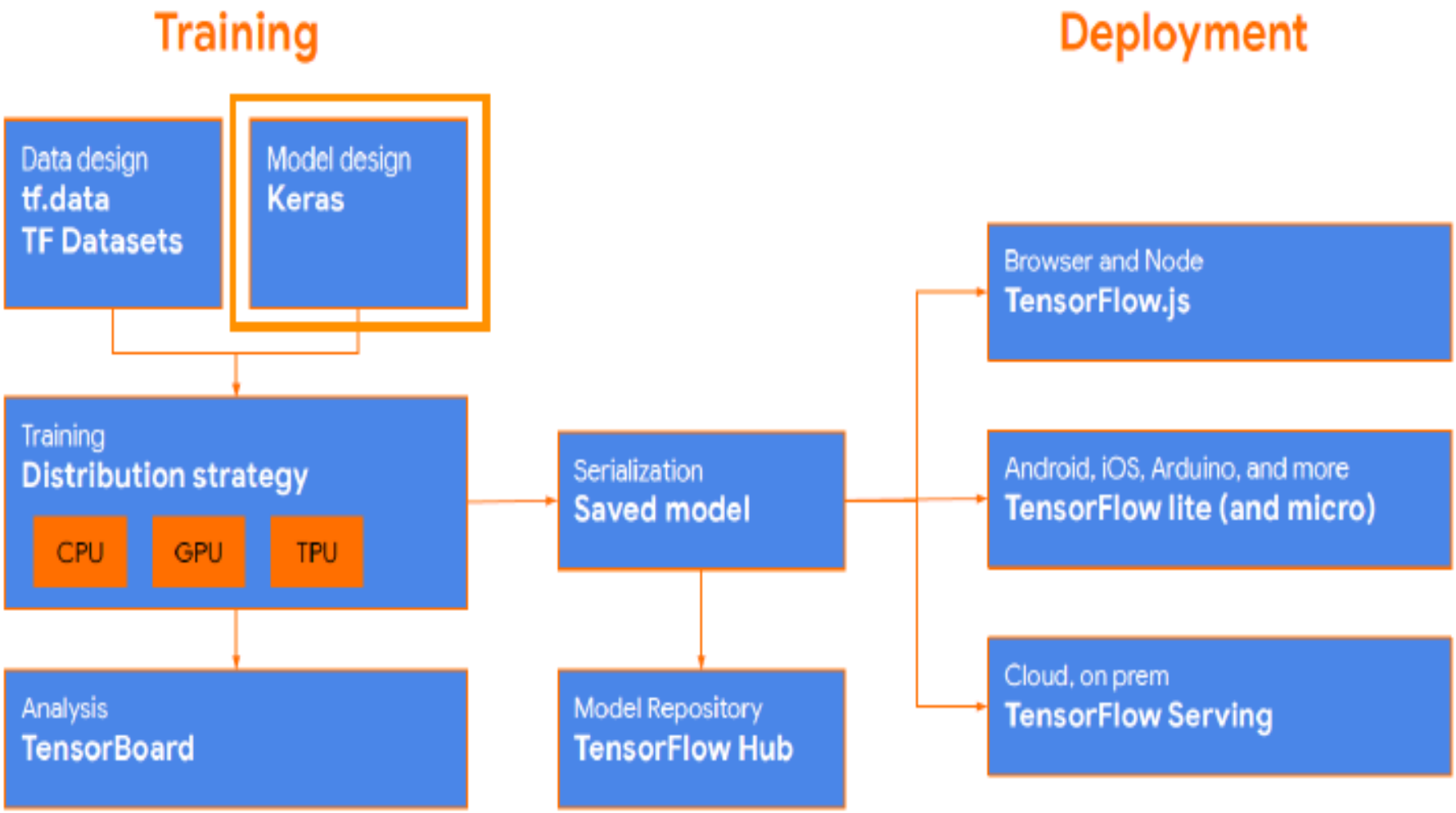
- Eager Execution (Define by Run 방식으로 바꿈)
- Session 이 없어지고 Function으로 대체
- AutoGraph
- Keras Python API
- API 정리함

Windows, macOS, Linux + Python (2.7), 3.x

```
$ pip install tensorflow
$ pip install tensorflow-gpu

$ pip install tensorflow==2.0.0
$ pip install tensorflow-gpu==2.0.0
```

텐서플로우 2

- TF 2의 개념도



텐서플로우 2

- Tensorflow 1.x

```
import tensorflow as tf
t= tf.nn.sigmoid([0.])
print(t)
  Tensor("Sigmoid_1:0", shape=(1,), dtype=float32)
```

- Tensorflow 2.x

```
import tensorflow as tf
t= tf.nn.sigmoid([0.])
print(t)
  tf.Tensor([0.5], shape=(1,), dtype=float32)
print(t.numpy())
  [0.5]
```

텐서플로우 2

- 텐서플로우 1과 2의 비교

TensorFlow 1.x

```
import tensorflow as tf

## 그래프를 정의합니다
g = tf.Graph()
with g.as_default():
    x = tf.placeholder(dtype=tf.float32,
                       shape=(None), name='x')
    w = tf.Variable(2.0, name='weight')
    b = tf.Variable(0.7, name='bias')
    z = w * x + b
    init = tf.global_variables_initializer()

## 세션을 만들고 그래프 g를 전달합니다
with tf.Session(graph=g) as sess:
    ## w와 b를 초기화합니다
    sess.run(init)
    ## z를 평가합니다
    for t in [1.0, 0.6, -1.8]:
        print('x=%4.1f --> z=%4.1f'%(
              t, sess.run(z, feed_dict={x:t})))
```

TensorFlow 2.x

```
import tensorflow as tf

w = tf.Variable(2.0, name='weight')
b = tf.Variable(0.7, name='bias')

# z를 평가합니다
for x in [1.0, 0.6, -1.8]:
    z = w * x + b
    print('x=%4.1f --> z=%4.1f'%(x, z))
```

텐서플로우 2

- AutoGraph

```
tf.Graph() + tf.Session() → @tf.function

# TensorFlow 1.x
output = session.run(ops, feed_dict={placeholder: input})


# TensorFlow 2.x
@tf.function
def simple_func():
    # complex computation with pure python
    ...
    return z
```

output=simple_func(input)

- for/while -> tf.while_loop
- if -> tf.cond

텐서플로우 2

- 그래프 정의

```
# 텐서플로 1.x 방식
g = tf.Graph()

# 그래프에 노드를 추가합니다.
with g.as_default():
...
```

```
# 텐서플로 2.x 방식
@tf.function
def simple_func():
    ...
    return z
```

텐서플로우 2

- 순차형 API (Sequential API)

[방법 1]

```
from tensorflow import tf

model = tf.keras.Sequential()
# 64개의 유닛을 가진 완전 연결 층을 모델에 추가합니다:
model.add(tf.keras.layers.Dense(64, activation='relu'))
# 또 하나를 추가합니다:
model.add(tf.keras.layers.Dense(64, activation='relu'))
# 10개의 출력 유닛을 가진 소프트맥스 층을 추가합니다:
model.add(tf.keras.layers.Dense(10, activation='softmax'))

# 컴파일
model.compile(optimizer=tf.keras.optimizers.Adam(0.001),
              loss='categorical_crossentropy',
              metrics=['accuracy'])

# 모델 훈련
model.fit(train_data, labels, epochs=10, batch_size=32)
# 모델 평가
model.evaluate(test_data, labels)
# 샘플 예측
model.predict(new_sample)
```

[방법2]

```
model = tf.keras.Sequential([
    tf.keras.layers.Dense(64),
    tf.keras.layers.Dense(64),
    tf.keras.layers.Dense(10),
])
```

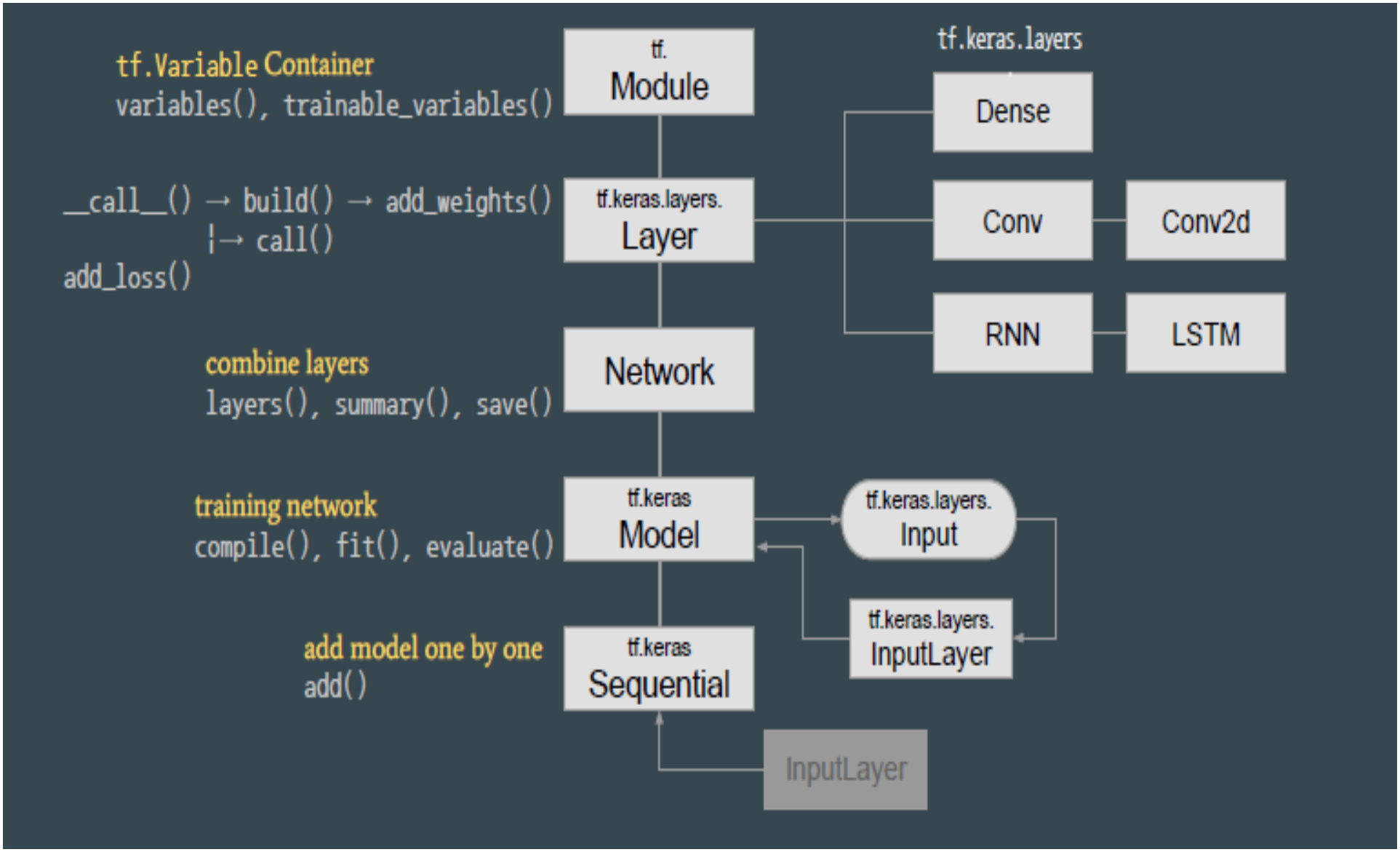
텐서플로우 2

- 함수형 API (Sequential API)

```
from tensorflow import tf

# 입력과 출력을 연결
input = tf.keras.Input(shape=(784,), name='img')
h1 = tf.keras.layers.Dense(64, activation='relu')(inputs)
h2 = tf.keras.layers.Dense(64, activation='relu')(h1)
output = tf.keras.layers.Dense(10, activation='softmax')(h2)

# 모델 생성
model = tf.keras.Model(input, output)

# 컴파일
...
# 훈련
...
```

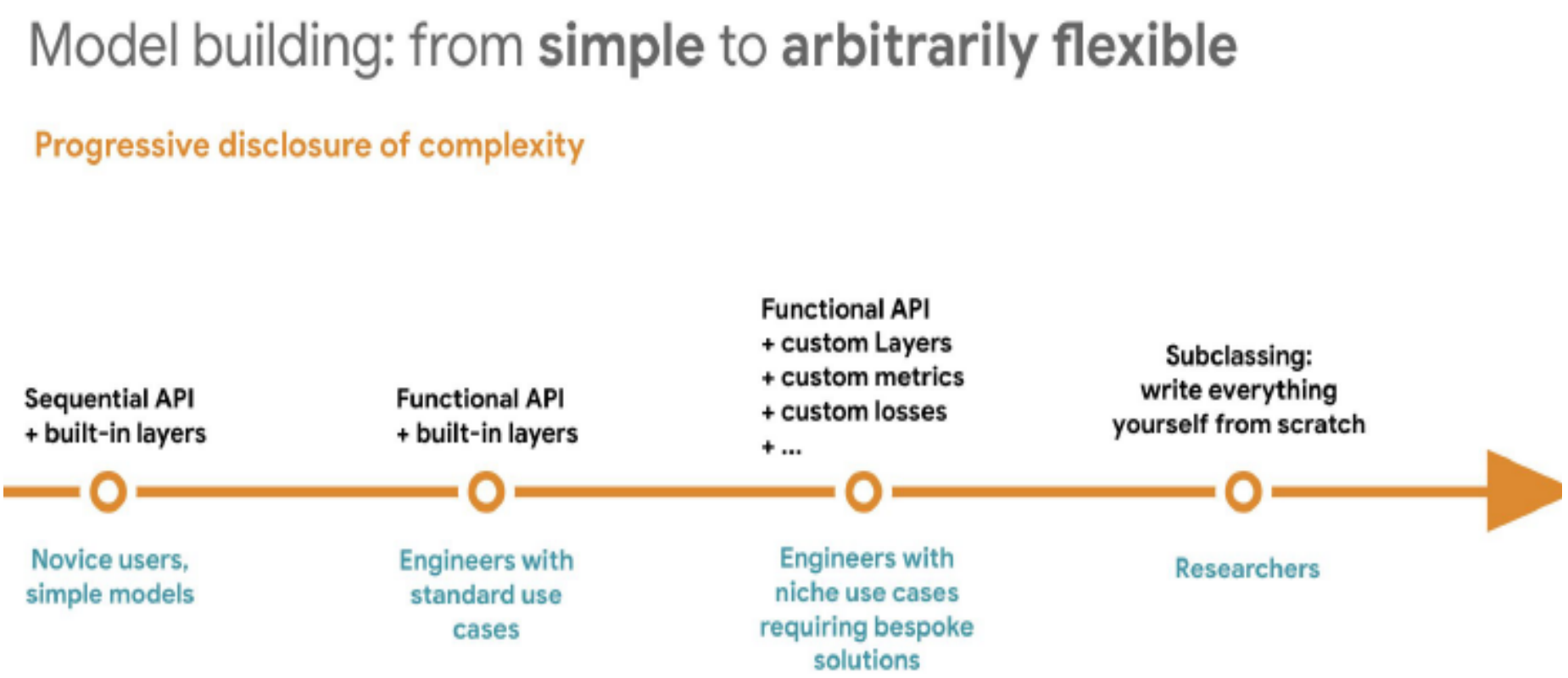
텐서플로우 2

- 클래스 계층



텐서플로우 2

- 모델 구축: 단순모델에서 매우 유연한 모델



텐서플로우 2

- 사용자 정의 모델

Custom Layer

```
class MyLayer(keras.layers.Layer):

    def __init__(self, units, activation=None, **kwargs):
        self.units = units
        self.activation = keras.activations.get(activation)
        super().__init__(**kwargs)

    def build(self, input_shape):
        self.kernel = self.add_weight(name='kernel',
                                      shape=(input_shape[1], self.units),
                                      initializer='uniform')
        self.bias = self.add_weight(name='bias',
                                    shape=(self.units,),
                                    initializer='zeros')
        super().build(input_shape)

    def call(self, X):
        z = tf.matmul(X, self.kernel) + self.bias
        return self.activation(z)
```

텐서플로우 2

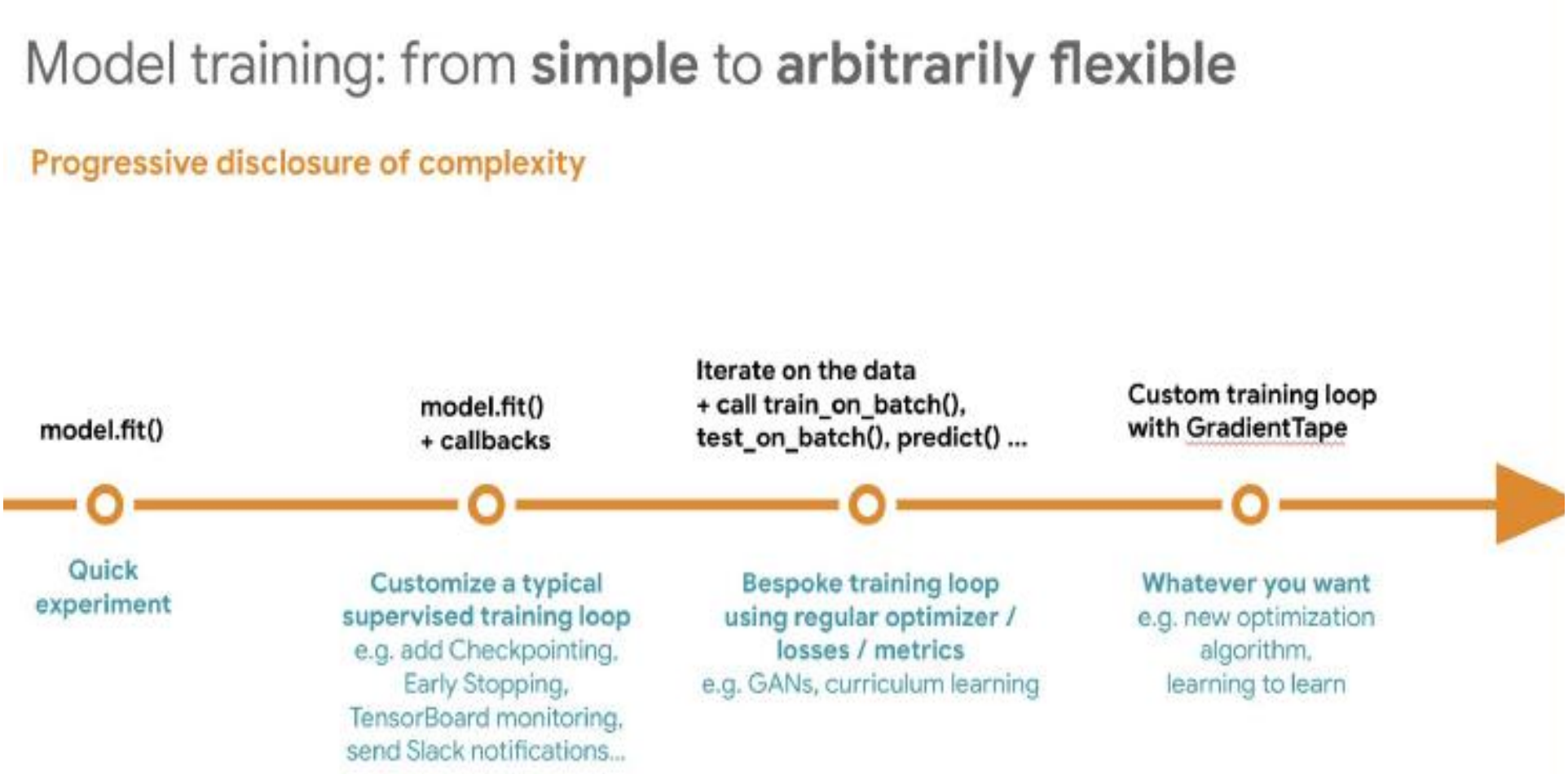
- 사용자 정의 모델

Custom Model

```
class MyModel(keras.models.Model):

    def __init__(self, **kwargs):
        self.hidden = MyLayer(10, activation="relu")
        self.output = MyLayer(1)
        super().__init__(**kwargs)

    def call(self, input):
        h = self.hidden(input)
        return self.output(h)

model = MyModel()
model.compile(...)
model.fit(...)
model.evaluate(...)
```

텐서플로우 2

- 모델 구축: 단순모델에서 매우 유연한 모델



텐서플로우 2

- model.fit()

```
model.compile(optimizer=Adam(),
              loss=BinaryCrossentropy(),
              metrics=[AUC(), Precision(), Recall()])


model.fit(data,
          epochs=10,
          validation_data=val_data,
          callbacks=[EarlyStopping(),
                     TensorBoard(),
                     ModelCheckpoint()])
```

...or write your own callbacks!

텐서플로우 2

- tf.GradientTape(): 자동미분 연산을 기록

```
@tf.function
def train_step(features, labels):
  with tf.GradientTape() as tape:
    logits = model(features, training=True)
    loss = loss_fn(labels, logits)

  grads = tape.gradient(loss, model.trainable_variables)
  optimizer.apply_gradients(zip(grads, model.trainable_variables))
  return loss
```

텐서플로우 2

- 권장사항
  - 케라스 층과 모델을 사용
  - 파이썬 함수와 같이 사용
  - @tf.function(autograph) 사용
  - tf.data.Datasets

텐서플로우 2 실습

- 코랩에서 실습 (1과 2)

1. 기본

https://colab.research.google.com/drive/1TjGh5OnR0WCTDQGzByrg7Ir3XrkxHxLH

2. 케라스창시자로부터배우는TensorFlow2.0 + Keras특강 (번역 박창선님)

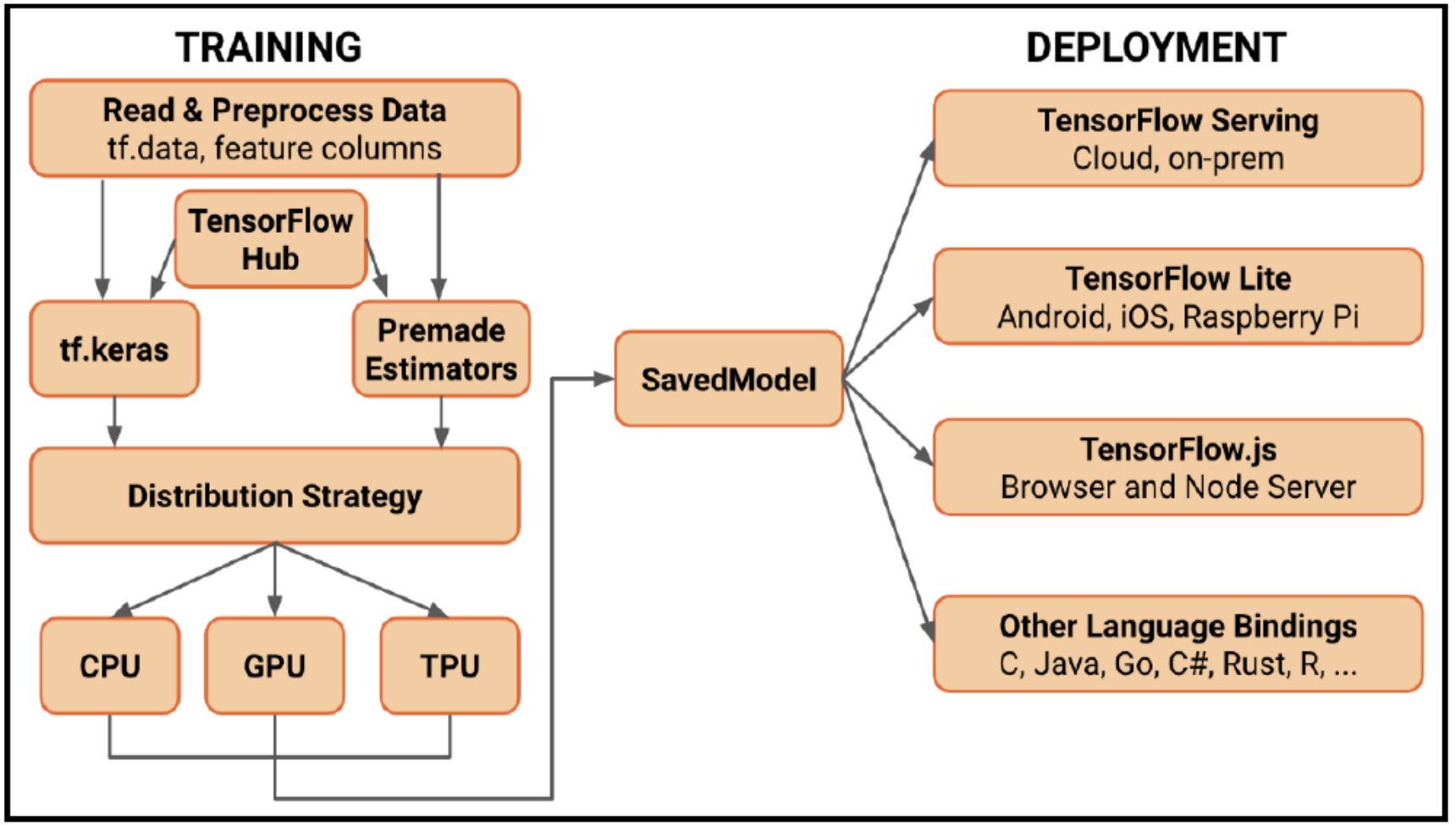
https://colab.research.google.com/drive/1tAmh9bN0-5CXm6H99uhhMpxprbRMMJoa

2-1. TensorFlow2.0 + KerasCrash Course (**François Chollet)**

https://colab.research.google.com/drive/1Fe_hd6g2rYnfjs1thyI0PC6aPYG9I1cN

텐서플로우 2

- SavedModel



The TensorFlow 2.0 training and deployment ecosystem. Image source: https://medium.com/tensorflow/whats-coming-in-tensorflow-2-0-d3663832e9b8—the TensorFlow Team