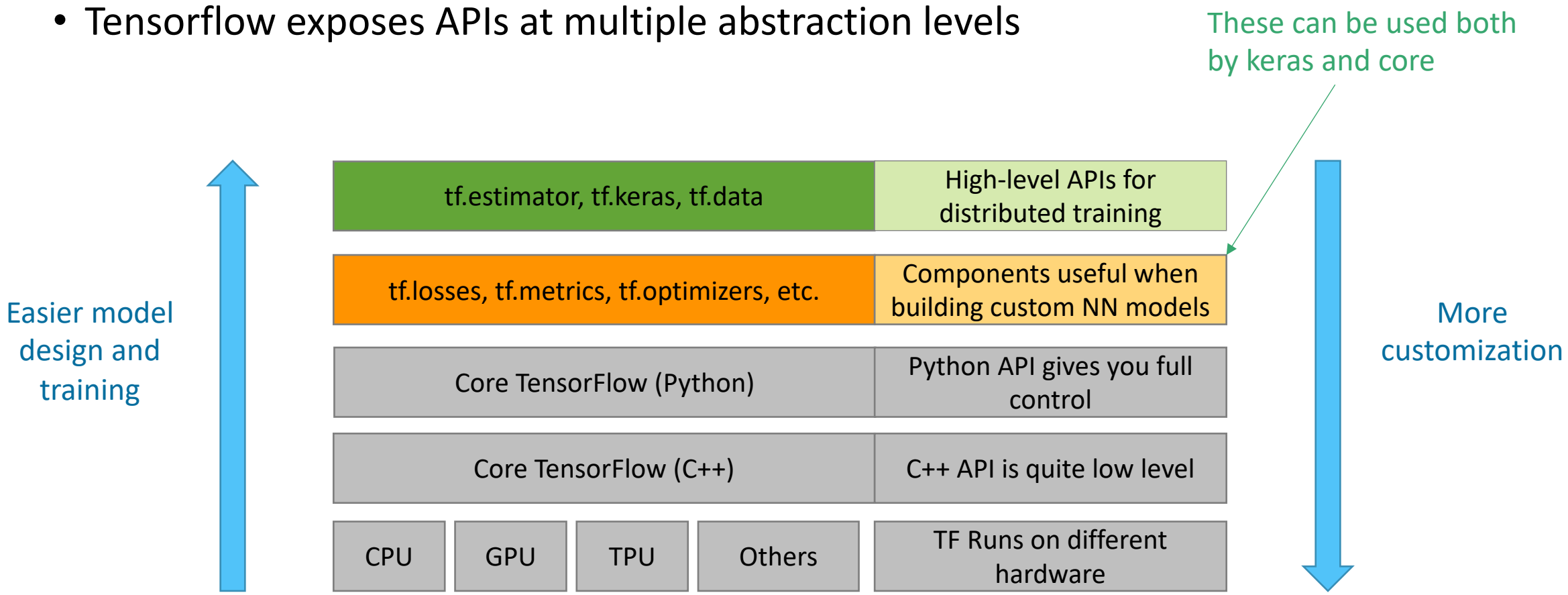
Tensorflow 2

Keras API

Tensorflow API Hierarchy

- Tensorflow exposes APIs at multiple abstraction levels



The Keras API

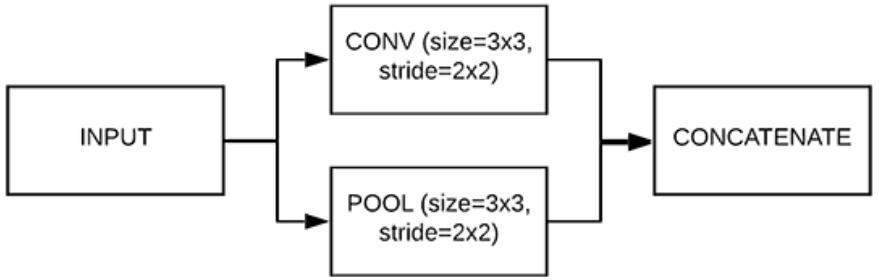
- Most of the times you don’t need to write custom models and trainign loops from scratch
- Keras is a high-level API built-in in TF2, that provides a much easier flow to write “standard” models
  - No need to worry about gradient tapes, weight updates, etc. manually.

- A Keras model can be built in 3 main ways:
  - **Sequential API** → for Single-Input Single-Output models built as stacks of layers
  - **Functional API** → for MIMO models, residual connections, etc.
  - **Sub-classing API** → for maximum customization (e.g. dynamic/adaptive models)

The Keras API

1. Sequential API

2. Functional API



3. Model Subclassing

```
tensorflow.keras.Model
    class MySimpleNN(Model):
        ...
```

Tensorflow 2

Keras Sequential API

The Keras Sequential API

1. Import Keras (the TF version):

```
import tensorflow as tf
from tensorflow import keras
```

2. Build the model:

```
model = keras.Sequential([
    keras.layers.Flatten(input_shape=(28, 28)),
    keras.layers.Dense(128, activation='relu'),
    keras.layers.Dense(10)
])
```

A layer that flattens a (batch of) rank-N tensors (N =2 in this case) to a rank-1 tensor

A stack of three layers

Two fully-connected layers (optionally with non-linear activation).
Internal weights stored as tf.Variable()

The Keras Sequential API

3. Compile the model:

```
model.compile(
  optimizer='adam',
  loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
  metrics=['accuracy']
)
```

Specify one of the many built-in gradient descent-based optimizers
Accepts either strings or tf.keras classes

(Adam)

Specify the metrics to monitor during training and testing

Specify a loss function. In this case:

- Categorical cross-entropy for multi-class classification
- `from_logits=True` because the last `Dense()` layer didn't include a softmax activation
- Sparse… because we encode labels as [0, 1, 2, …] and not using one-hot encoding

The Keras Sequential API

4. Train the model:

```
history = model.fit(X_train, Y_train, epochs=10)
```

Automatically takes care of recording computations with tapes, updating gradients, etc.

A data structure containing the training history (loss and metrics), useful to plot learning curves etc.

e.g. history.history['loss'] contains the loss in each epoch.

The Keras Sequential API

5. Test the model:

```
test_loss, test_acc = model.evaluate(X_test, Y_test)
```

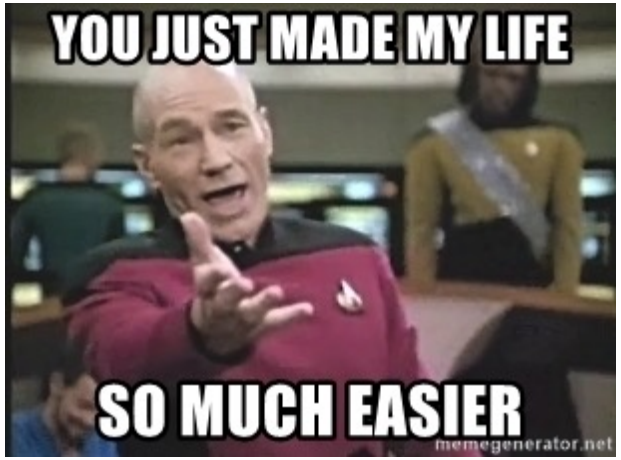
6. Use the model for new predictions:

```
pred = model.predict(X_test)
class_first = numpy.argmax(pred[0])
```

The Keras Sequential API

- Summary:

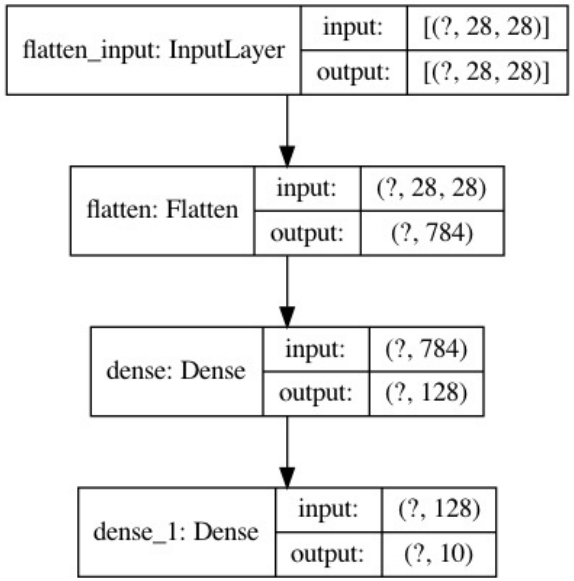
```
1  import tensorflow as tf
   from tensorflow import keras

   # get some data to train on… (see later)

2  model = keras.Sequential([
       keras.layers.Flatten(input_shape=(28, 28)),
       keras.layers.Dense(128, activation='relu'),
       keras.layers.Dense(10)
   ])

3  model.compile(optimizer='adam',
   loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
   metrics=['accuracy'])

4  history = model.fit(X_train, Y_train, epochs=10)
5  test_loss, test_acc = model.evaluate(X_test, Y_test)
6  pred = model.predict(X_test)
```



The Keras Sequential API

- After creating a model, you can also see a summary of its layers with `model.summary()`

```
In [5]: model.summary()
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
flatten (Flatten)            (None, 784)               0
_________________________________________________________________
dense (Dense)                (None, 128)               100480
_________________________________________________________________
dense_1 (Dense)              (None, 10)                1290
=================================================================
Total params: 101,770
Trainable params: 101,770
Non-trainable params: 0
_________________________________________________________________
```



- Or plot it using:

```
keras.utils.plot_model(model, <filename>,
show_shapes=True)
```

The Keras Sequential API

- **Notebook:** Keras_Sequential_API.ipynb