Kissipo Learning for Deep Learning

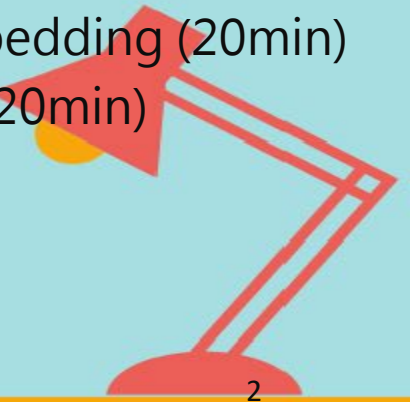
Topic 11: TensorFlow overview (20min)

Hsueh-Ting Chu

KLDL-W4-11

Topics

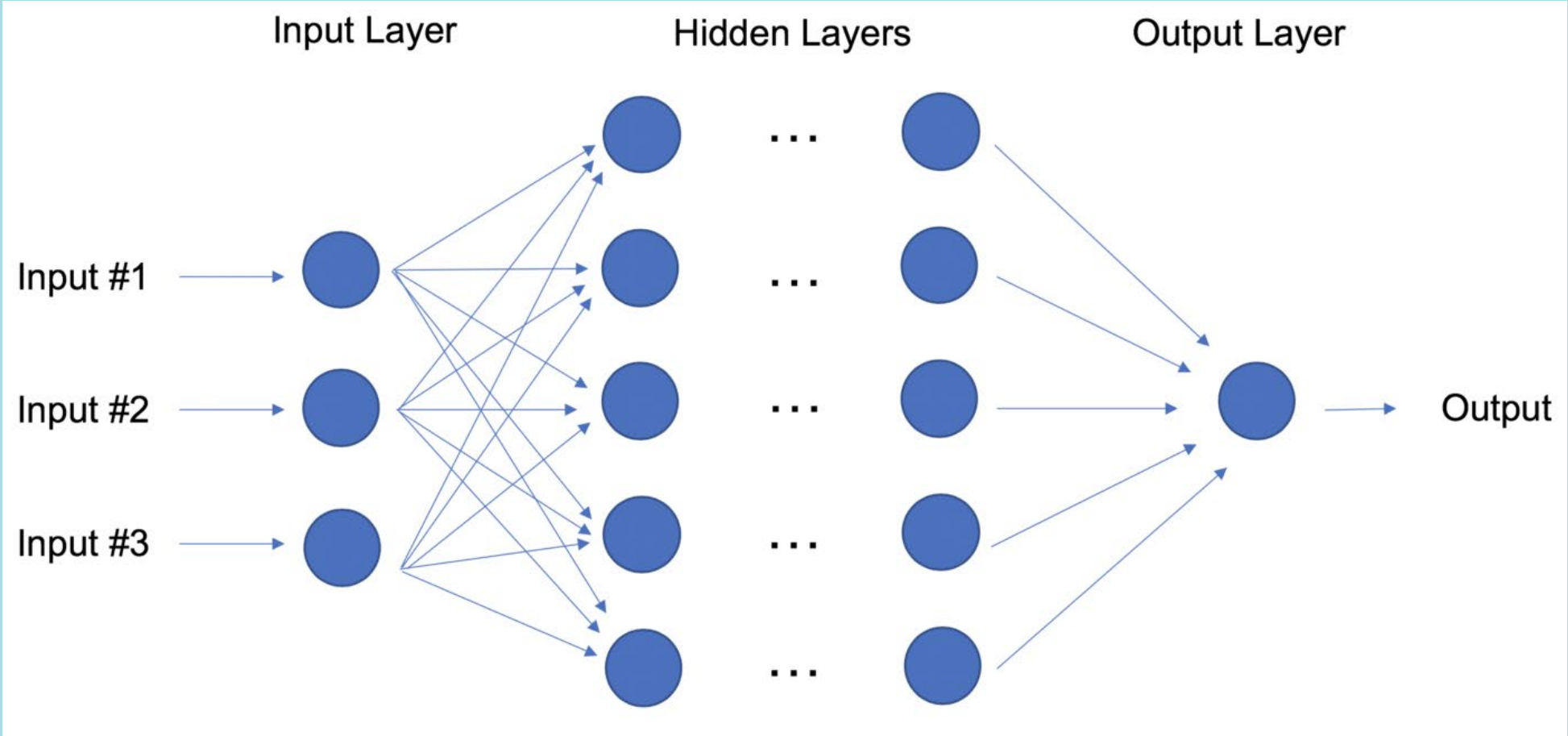
- Topic 01: Introduction to Deep Learning (20min)
- Topic 02: Kissipo Learning for Deep Learning (20min)
- Topic 03: Python quick tutorial (20min)
- Topic 04: Numpy quick tutorial (15min)
- Topic 05: Pandas quick tutorial (15min)
- Topic 06: Scikit-learn quick tutorial (15min)
- Topic 07: OpenCV quick tutorial (15min)
- Topic 08: Image Processing basics (20min)
- Topic 09: Machine Learning basics (20min)
- Topic 10: Deep Learning basics (20min)
- Topic 11: TensorFlow overview (20min)
- Topic 12: CNN with TensorFlow (20min)
- Topic 13: RNN with TensorFlow (20min)
- Topic 14: PyTorch overview (20min)
- Topic 15: CNN with PyTorch (20min)
- Topic 16: RNN with Pytorch (20min)
- Topic 17: Introduction to AOI (20min)
- Topic 18: AOI simple Pipeline (A) (20min)
- Topic 19: AOI simple Pipeline (B) (20min)
- Topic 20: Introduction to Object detection (20min)
- Topic 21: YoloV5 Quick Tutorial (20min)
- Topic 22: Using YoloV5 for RSD (20min)
- Topic 23: Introduction to NLP (20min)
- Topic 24: Introduction to Word Embedding (20min)
- Topic 25: Name prediction project (20min)



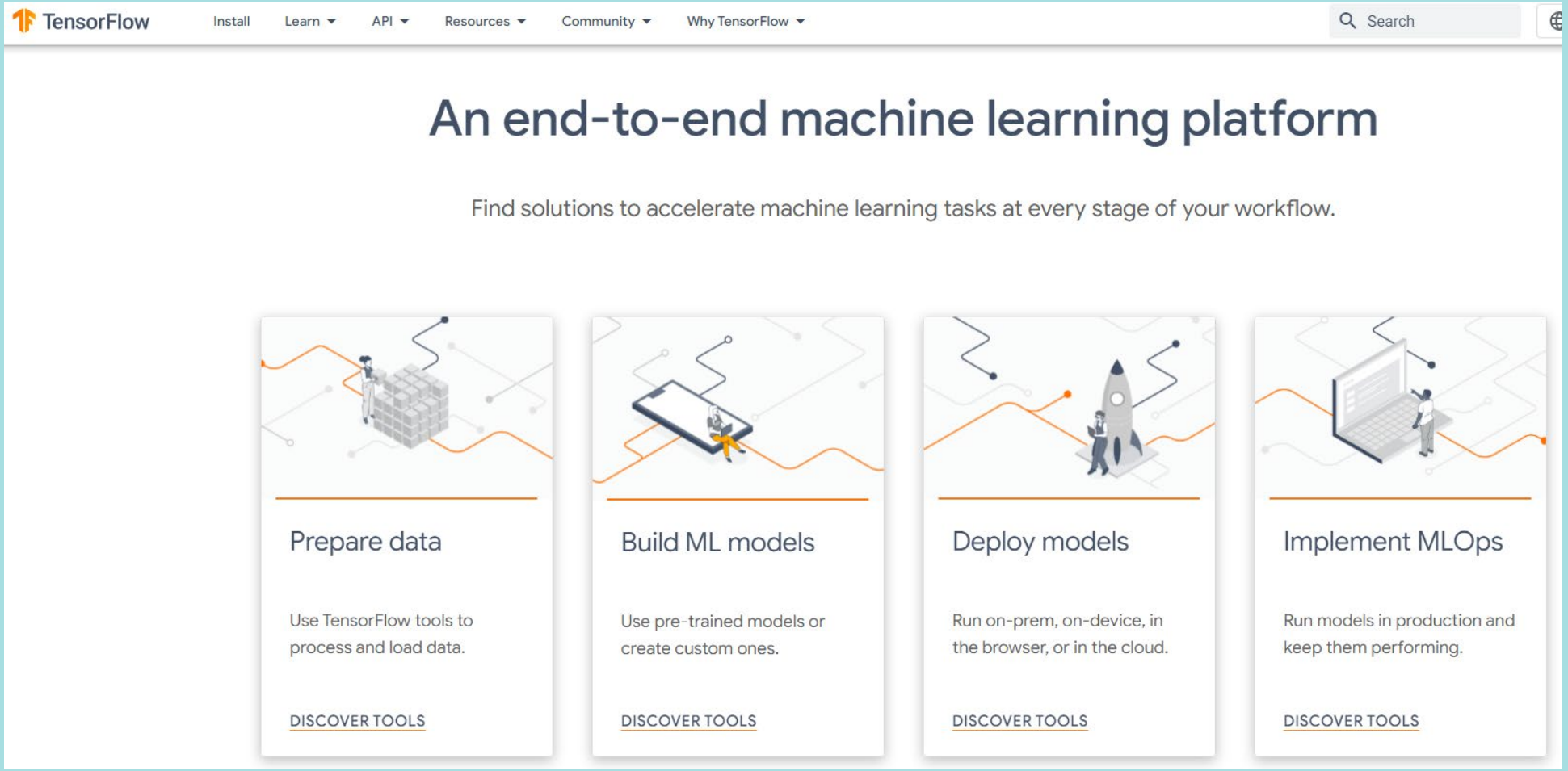
Content

- Topic 11: TensorFlow overview (20min)
  - TensorFlow as a platform
  - TensorFlow as a SDK
  - TensorFlow 2.0 three model APIs
  - Training a model for MNIST handwritten digit dataset

About deep learning (Neural Network)



TensorFlow as a platform



https://www.tensorflow.org/

TensorFlow as a SDK

For beginners

The best place to start is with the user-friendly Sequential API. You can create models by plugging together building blocks. Run the "Hello World" example below, then visit the tutorials to learn more.

To learn ML, check out our education page . Begin with curated curriculums to improve your skills in foundational ML areas.

```
import tensorflow as tf
mnist = tf.keras.datasets.mnist

(x_train, y_train),(x_test, y_test) = mnist.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0

model = tf.keras.models.Sequential([
  tf.keras.layers.Flatten(input_shape=(28, 28)),
  tf.keras.layers.Dense(128, activation='relu'),
  tf.keras.layers.Dropout(0.2),
  tf.keras.layers.Dense(10, activation='softmax')
])

model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

model.fit(x_train, y_train, epochs=5)
model.evaluate(x_test, y_test)
```

For experts

The Subclassing API provides a define-by-run interface for advanced research. Create a class for your model, then write the forward pass imperatively. Easily author custom layers, activations, and training loops. Run the "Hello World" example below, then visit the tutorials to learn more.

```
class MyModel(tf.keras.Model):
  def __init__(self):
    super(MyModel, self).__init__()
    self.conv1 = Conv2D(32, 3, activation='relu')
    self.flatten = Flatten()
    self.d1 = Dense(128, activation='relu')
    self.d2 = Dense(10, activation='softmax')

  def call(self, x):
    x = self.conv1(x)
    x = self.flatten(x)
    x = self.d1(x)
    return self.d2(x)
model = MyModel()

with tf.GradientTape() as tape:
  logits = model(images)
  loss_value = loss(logits, labels)
grads = tape.gradient(loss_value, model.trainable_variable
optimizer.apply_gradients(zip(grads, model.trainable_varia
```

https://www.tensorflow.org/

TensorFlow 2.0 three model APIs

(1) Sequential API

(2) Subclassing API

(3) Functional API

Sequential model

TensorFlow Core

總覽 教學課程 指南 TF 1↗

TensorFlow 教學課程
適合新手的快速入門導覽課程
適合專家的快速入門導覽課程

新手

使用 Keras 進行機器學習的基本知識

載入及預先處理資料

Estimator

進階

自訂

分散式訓練

RSVP for your your local TensorFlow Everywhere event today! Find an event

TensorFlow > 學習 > TensorFlow Core > 教學課程

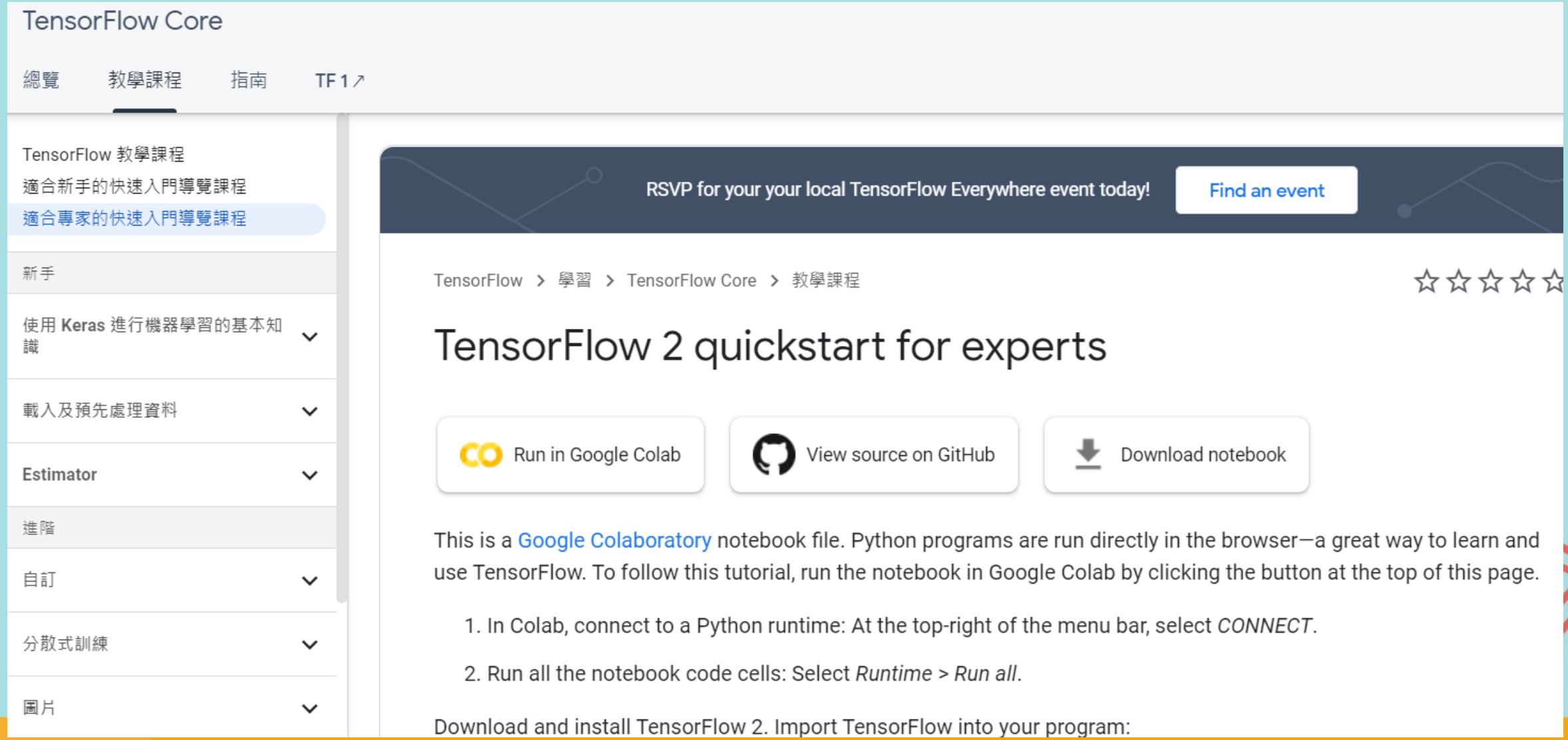
TensorFlow 2 quickstart for beginners

Run in Google Colab | View source on GitHub | Download notebook

This short introduction uses Keras to:

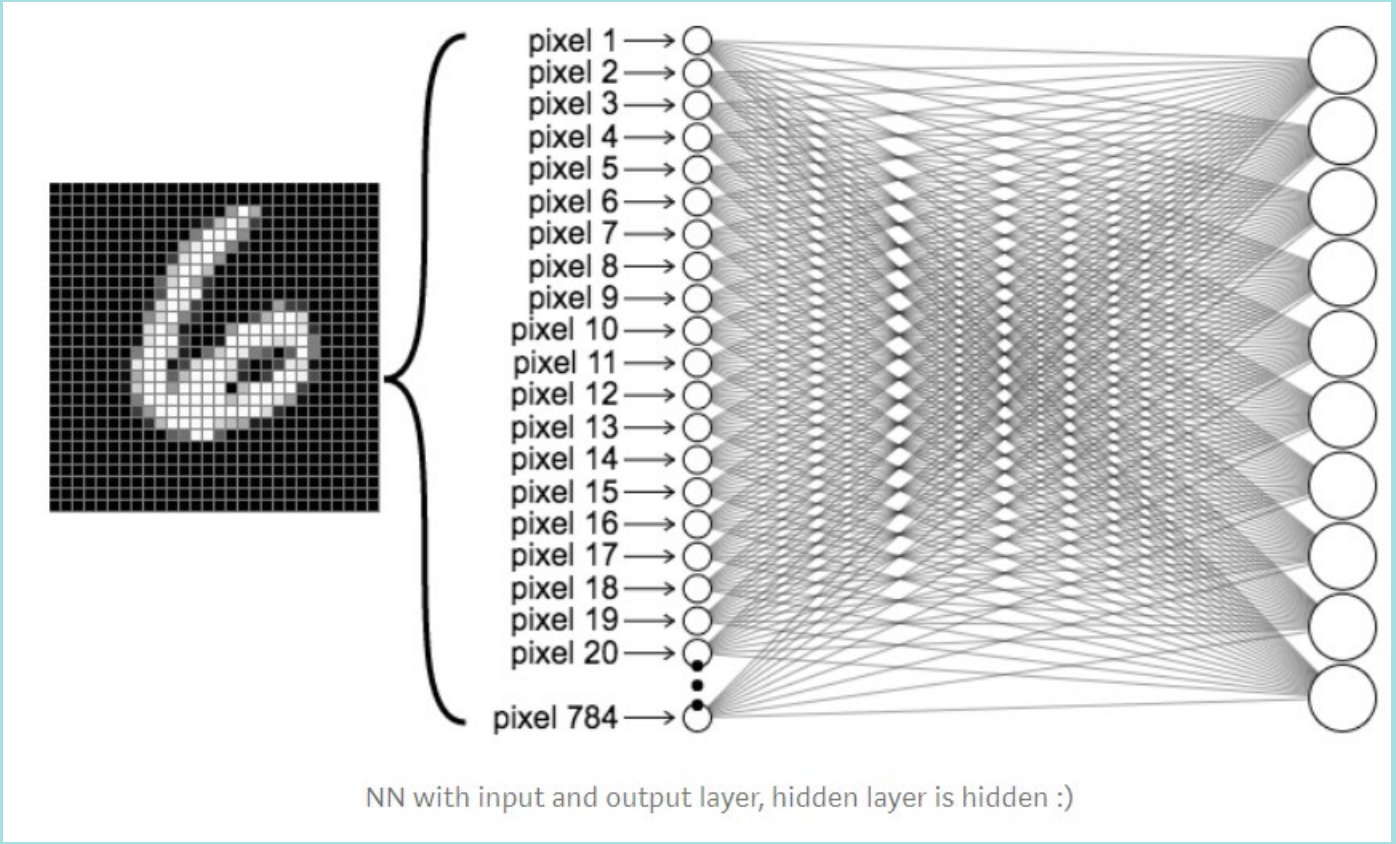
1. Build a neural network that classifies images.
2. Train this neural network.
3. And, finally, evaluate the accuracy of the model.

Model subclassing



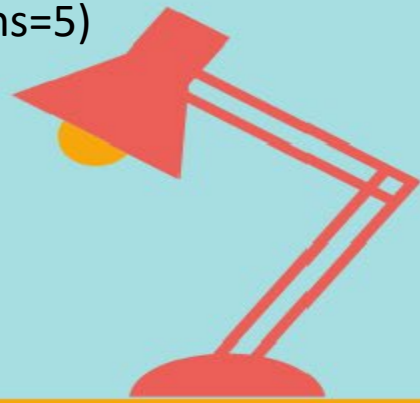
MNIST handwritten digit dataset

Feed Forward Neural Networks



NN with input and output layer, hidden layer is hidden :)

```
model = keras.models.Sequential([
  keras.layers.Flatten(input_shape=(28, 28)),
  keras.layers.Dense(128, activation='relu'),
  keras.layers.Dropout(0.2),
  keras.layers.Dense(10, activation='softmax')
])

model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

model.fit(x_train, y_train, epochs=5)

model.evaluate(x_test, y_test)
```



IPO (1)-Hello Word

```
[2]  mnist = tf.keras.datasets.mnist
     (x_train, y_train), (x_test, y_test) = mnist.load_data()
     x_train, x_test = x_train / 255.0, x_test / 255.0
```

```
[3]  print(x_train.shape)
     print(y_train.shape)
```

```
     (60000, 28, 28)
     (60000,)
```

Input

Output

```
[4]  model = tf.keras.models.Sequential([
       tf.keras.layers.Flatten(input_shape=(28, 28)),
       tf.keras.layers.Dense(128, activation='relu'),
       tf.keras.layers.Dropout(0.2),
       tf.keras.layers.Dense(10)
     ])
```

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

Build a model

```
model = tf.keras.models.Sequential([
  tf.keras.layers.Flatten(input_shape=(28, 28)),
  tf.keras.layers.Dense(128, activation='relu'),
  tf.keras.layers.Dropout(0.2),
  tf.keras.layers.Dense(10, activation='softmax')
])
```

Training & Inference

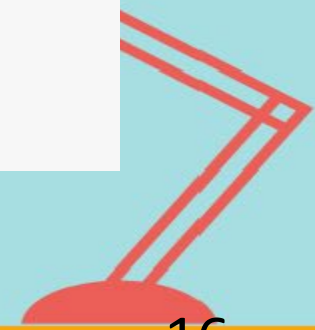
```
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

model.fit(x_train, y_train, epochs=5)

model.predict(test_img)
model.evaluate(x_test, y_test)
```

Test a single image

```
[ ]  from PIL import Image
     from IPython.display import display

[ ]  img = Image.open( "Digit4.bmp" )
     print(img.format, img.size, img.mode)

[ ]  display(img)

[ ]  import tensorflow as tf

[ ]  model = tf.keras.models.load_model('my_model.h5')
     model.summary()
```

```
[ ]  import numpy as np
     img = np.resize(img, (28,28))
     im2arr = np.array(img)
     im2arr = im2arr.reshape(1, 28,28)
     y_pred = new_model.predict_classes(im2arr)
     print(y_pred)

[ ]  img = Image.open( "DigitX.bmp" )
     print(img.format, img.size, img.mode)
     display(img)
```

```
     img = np.resize(img, (28,28))
     im2arr = np.array(img)
     im2arr = im2arr.reshape(1, 28,28)
     y_pred = new_model.predict_classes(im2arr)
     print(y_pred)
```



Thanks!

Q&A