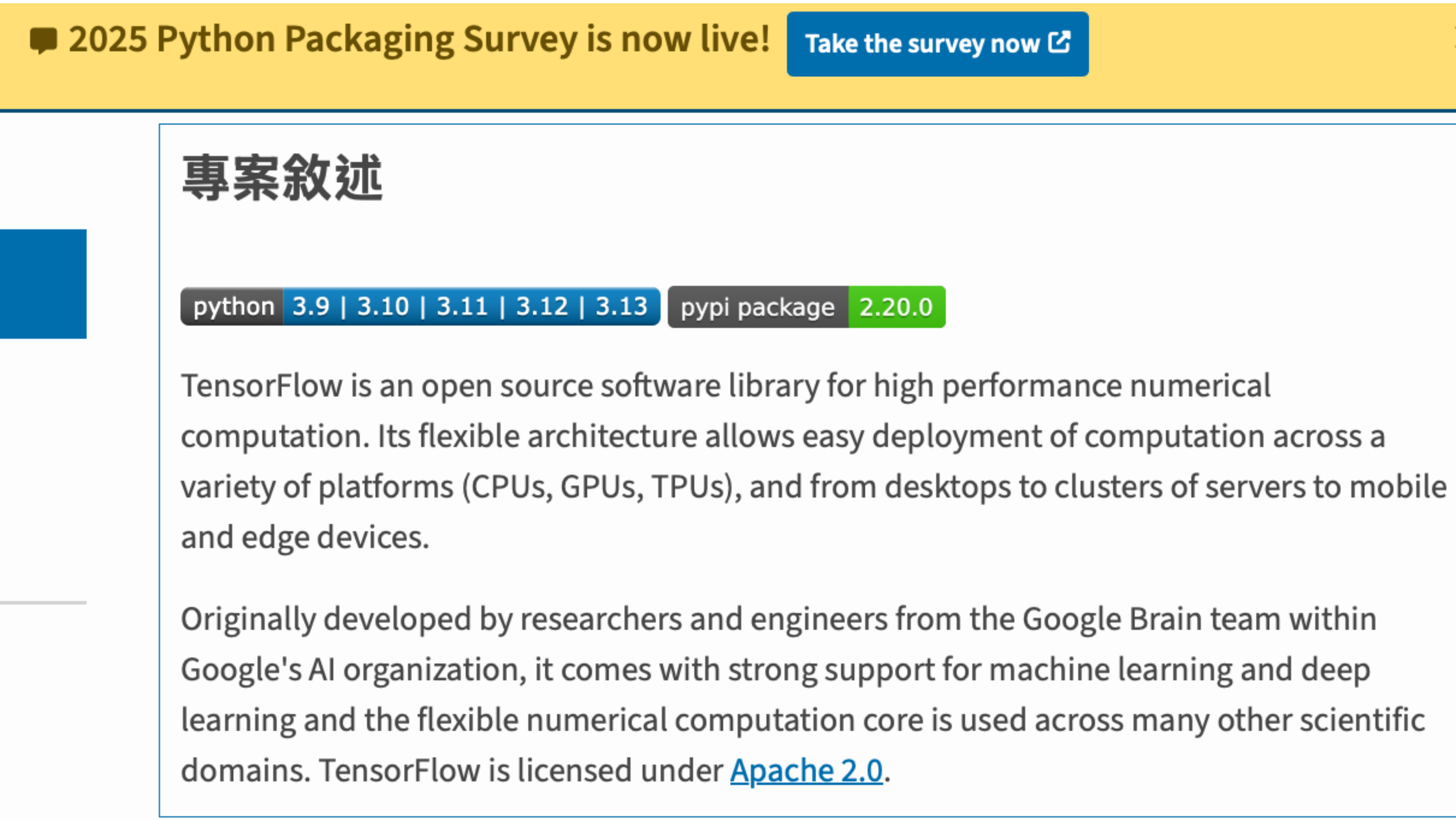
Tensor Flow for computation of gradients

Tensor flow


2025 Python Packaging Survey is now live!
Take the survey now
python 3.9 | 3.10 | 3.11 | 3.12 | 3.13
pypi package 2.20.0

專案敘述

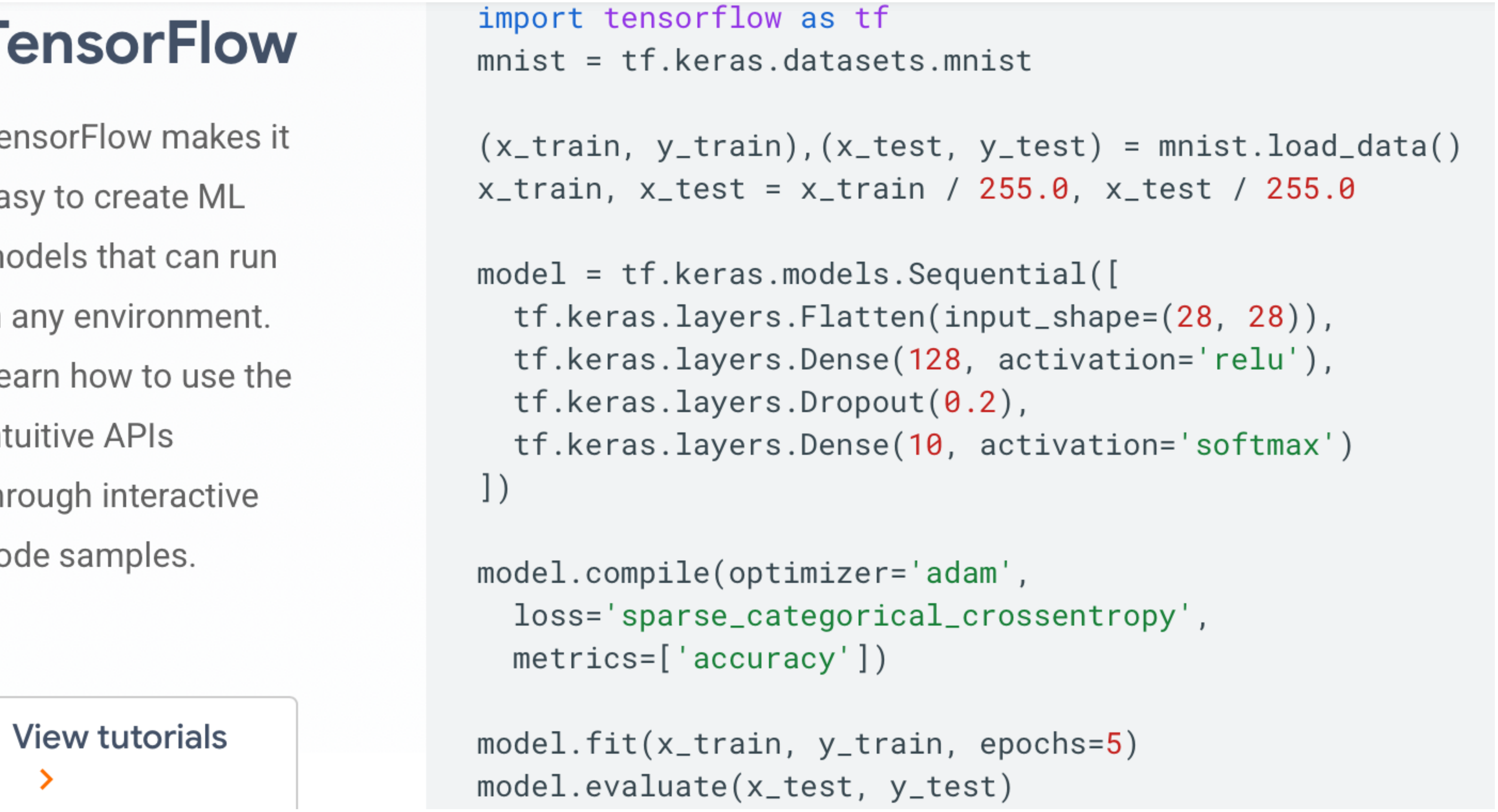
TensorFlow is an open source software library for high performance numerical computation. Its flexible architecture allows easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs), and from desktops to clusters of servers to mobile and edge devices.

Originally developed by researchers and engineers from the Google Brain team within Google's AI organization, it comes with strong support for machine learning and deep learning and the flexible numerical computation core is used across many other scientific domains. TensorFlow is licensed under Apache 2.0.

★ 193k TF 2.20 released

端對端機器學習平台

安裝 TensorFlow



ensorFlow

ensorFlow makes it
asy to create ML
odels that can run
 any environment.
earn how to use the
tuitive APIs
rough interactive
ode samples.

View tutorials
›

```python
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)
```

Example 1

$$\begin{tikzcd}
\boxed{x=2} \arrow[r] & x^2 \arrow[dr] & & \\
 & & \sum \arrow[r] & \boxed{z} \\
\boxed{y=3} \arrow[r] & tanh(y^2) \arrow[ur] & &
\end{tikzcd}$$

$$z = ?$$

$$\frac{dz}{dx}\Big|_{x=2,y=3} = ? \qquad \frac{dz}{dy}\Big|_{x=2,y=3} = ?$$

Example 1

$$\begin{tikzcd}
\boxed{x=2} \arrow[r] & x^2 \arrow[dr] & & \\
& & \sum \arrow[r] & \boxed{z} \\
\boxed{y=3} \arrow[r] & tanh(y^2) \arrow[ur] & &
\end{tikzcd}$$

```python
x = tf.Variable(2.0)
y = tf.Variable(3.0)

with tf.GradientTape() as t:
   y_sq = tf.math.tanh(y**2)
   z = x**2 + y_sq

grad = t.gradient(z, {'x': x, 'y': y})

print('dz/dx:', grad['x'])  # 2*x => 4
print('dz/dy:', grad['y'])
```

Example 2

$$\begin{tikzcd} \boxed{x=1} \arrow[r] & x^3 \arrow[r] & \boxed{y} \end{tikzcd}$$

$$\frac{dy}{dx}\Big|_{x=1} = ?$$

$$\frac{d^2y}{dx^2}\Big|_{x=1} = ?$$

Example 2

```python
x = tf.Variable(1.0)  # Create a Tensorflow variable initialized to
1.0

with tf.GradientTape() as t2:
  with tf.GradientTape() as t1:
    y = x * x * x

  # Compute the gradient inside the outer `t2` context manager
  # which means the gradient computation is differentiable as well.
  dy_dx = t1.gradient(y, x)
d2y_dx2 = t2.gradient(dy_dx, x)

print('dy_dx:', dy_dx.numpy())  # 3 * x**2 => 3.0
print('d2y_dx2:', d2y_dx2.numpy())  # 6 * x => 6.0
```

$$\begin{tikzcd}\boxed{x=1} \arrow[r] & x^3 \arrow[r] & \boxed{y}\end{tikzcd}$$

Example 3

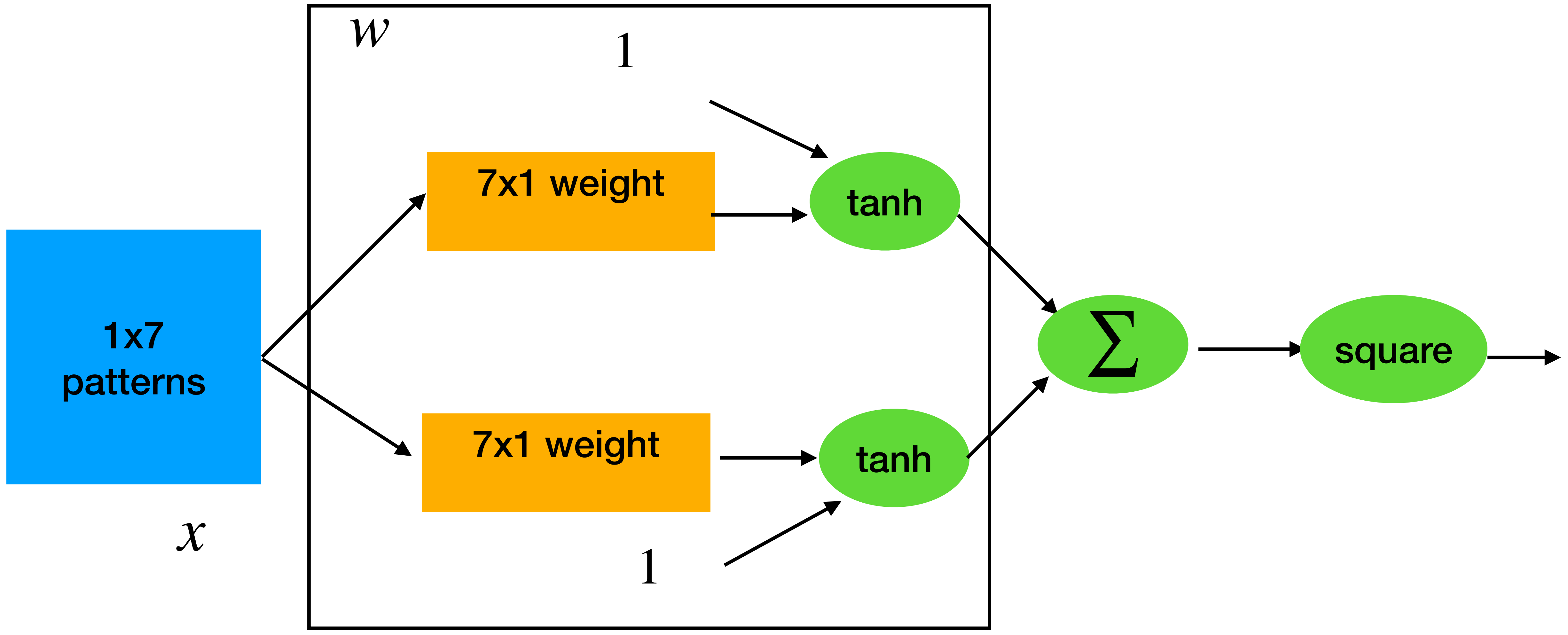
$$\begin{tikzcd}
x=[[1,2,3,4,5,6,7]] \arrow[r] & xw+b \arrow[r] & tanh \arrow[r] & \boxed{mean(y^2)} \arrow[d] \\
 & & & \boxed{l}
\end{tikzcd}$$

For w_{7x2} and b_{1x2}, $\dfrac{dl}{db} = ?$

$$\frac{dl}{dw} = ?$$

Example 3

$$\begin{tikzcd}
\text{x=[[1,2,3,4,5,6,7]]} \arrow[r] & xw + b \arrow[r] & tanh \arrow[r] & \boxed{mean(y^2)} \\
w_{7x2} \arrow[ur] & & b_{1x2} \arrow[ul] &
\end{tikzcd}$$



Example 3

$$\begin{tikzcd} \text{x=[[1,2,3,4,5,6,7]]} \arrow[r] & xw + b \arrow[r] & tanh \arrow[r] & \boxed{mean(y^2)} \end{tikzcd}$$

w_{7x2}

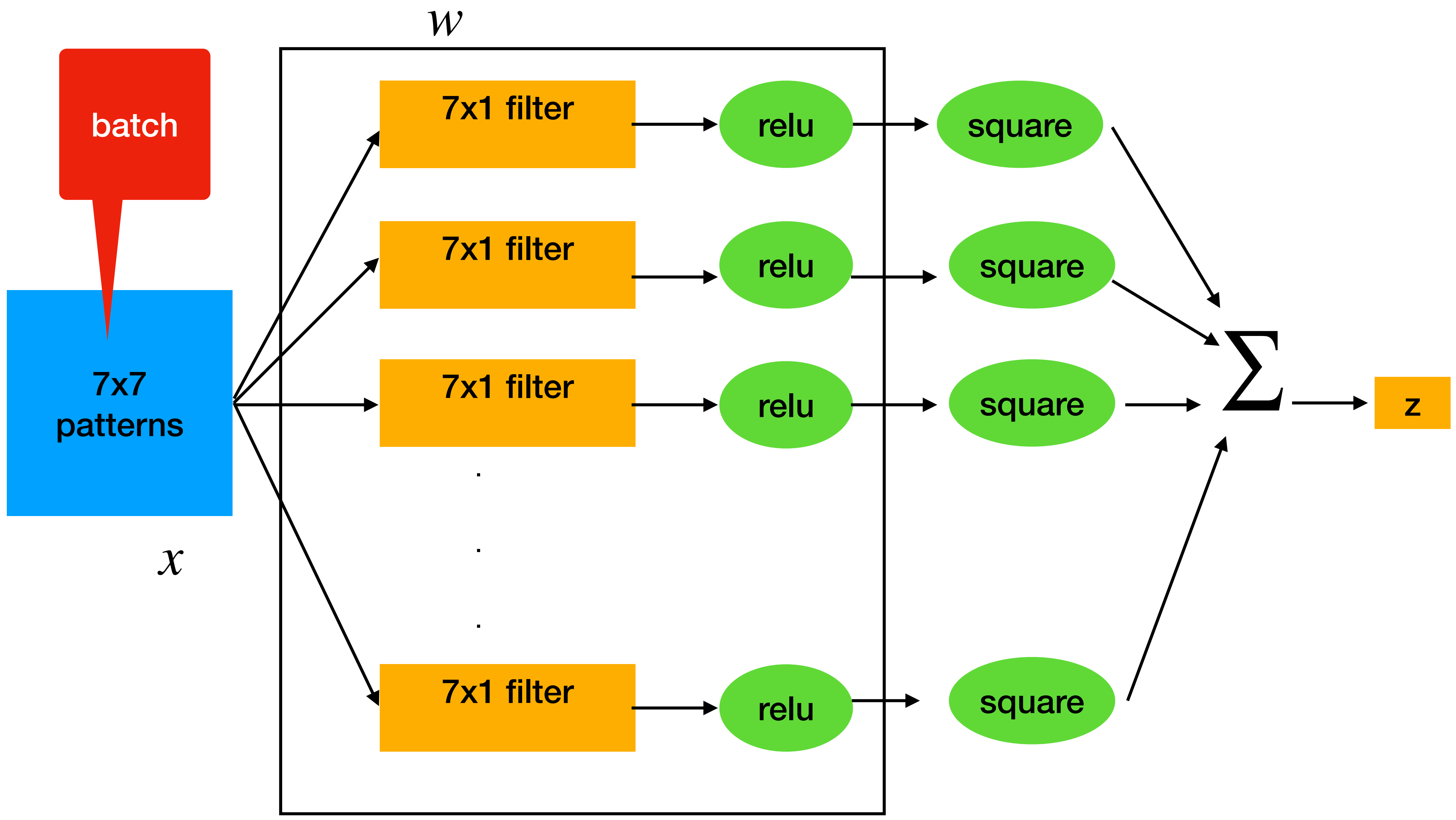
```
w = tf.Variable(tf.random.normal((7, 2)), name='w')
b = tf.Variable(tf.zeros(2, dtype=tf.float32), name='b')
x = [[1.,2.,3.,4.,5.,6.,7.]]
print(x[0:5])
with tf.GradientTape(persistent=True) as tape:
  h = x @ w + b
  y = tf.math.tanh(h)
  loss = tf.reduce_mean(y**2)
[dl_dw, dl_db] = tape.gradient(loss, [w, b])
print(w.shape)
print(dl_dw.shape)
print(dl_dw[0,:])
```

b_{1x2}

Example 4

$$l = norm(\frac{dz}{dx})$$

$$\frac{dl}{dw} = ?$$



Example 4

```
x = tf.random.normal([7, 7])

layer = tf.keras.layers.Dense(10, activation=tf.nn.relu)
with tf.GradientTape() as t2:
  # The inner tape only takes the gradient with respect to the inpu
  # not the variables.
  with tf.GradientTape(watch_accessed_variables=False) as t1:
    t1.watch(x)
    y = layer(x)
    out = tf.reduce_sum(layer(x)**2)
  # 1. Calculate the input gradient.
  g1 = t1.gradient(out, x)

  # 2. Calculate the magnitude of the input gradient.
  g1_mag = tf.norm(g1)

# 3. Calculate the gradient of the magnitude with respect to the 
model.
dg1_mag = t2.gradient(g1_mag, layer.trainable_variables)
print('dg1_mag:', dg1_mag)
```