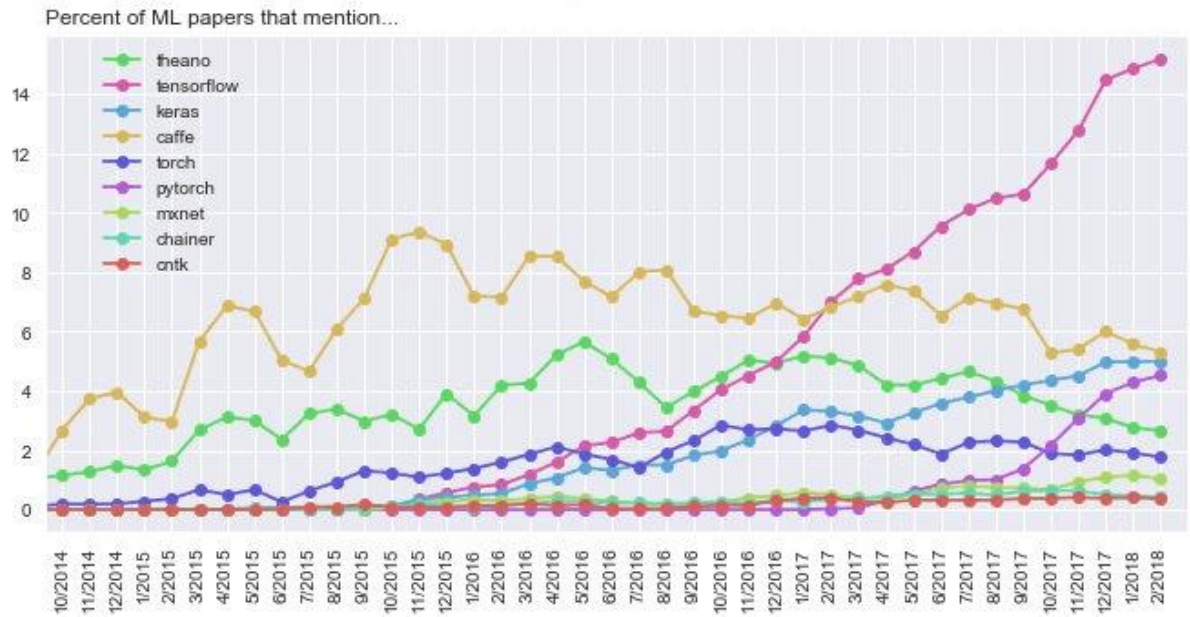
Keras és TensorFlow

Kertész Gábor

kertesz.gabor@sztaki.hu

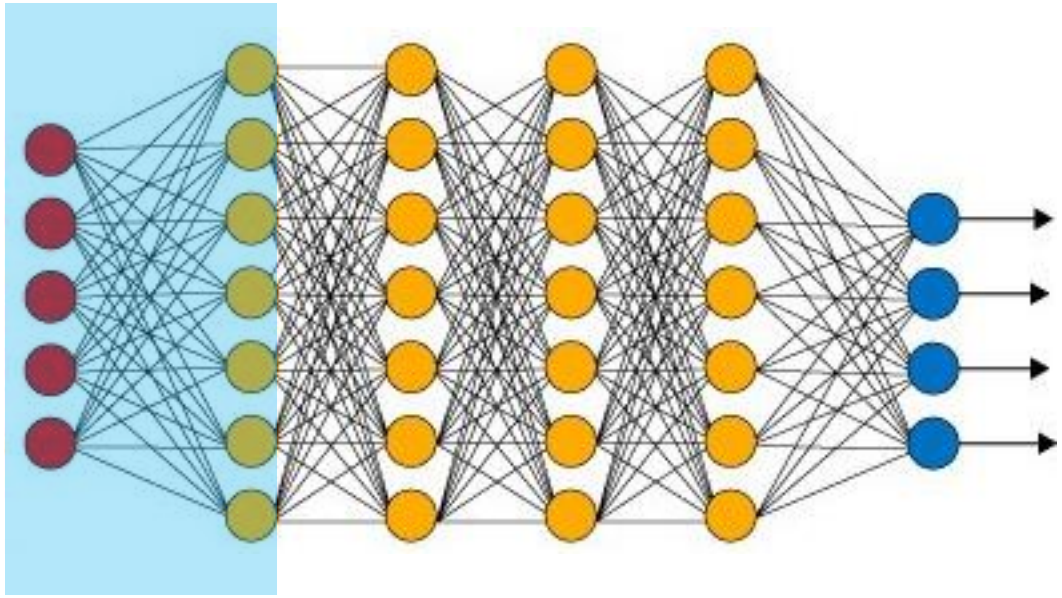
TensorFlow

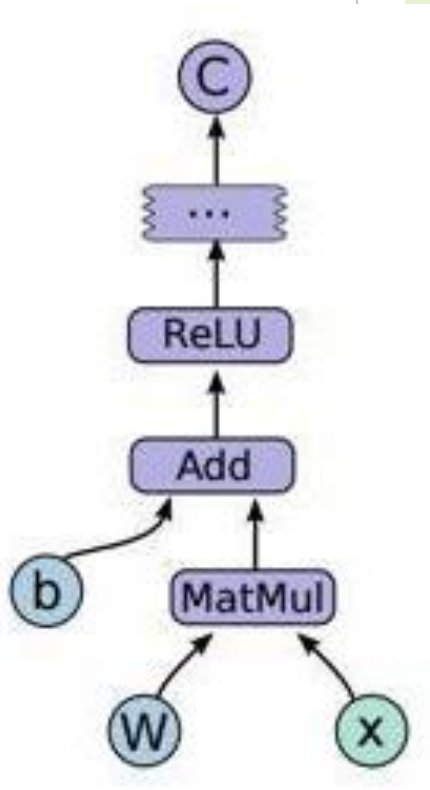
- Google Brain csapat által fejlesztett Python (C++, JS) függvénykönyvtár adatfolyam programozáshoz
- 2015 novemberében publikálták először
- CPU/GPU támogatás, több platformon



TensorFlow működése

- A számításokat egy adatfolyam gráfban kell elvégezni





TensorFlow működése

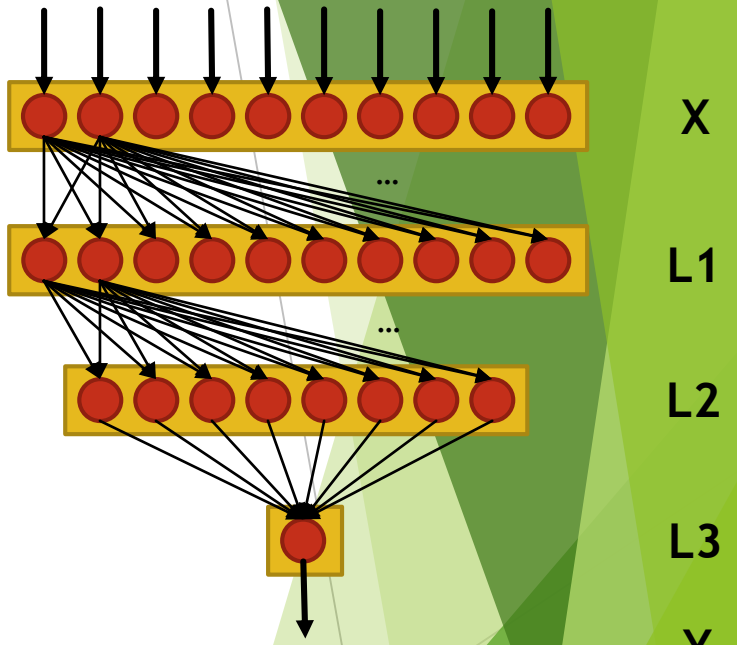
- A gráf definiálását követően a változókat inicializálni szükséges
- Tanításkor a hiba visszaterjesztéséhez használt gradiens a műveleti gráf alapján könnyen kiszámítható
- A veszteségfüggvény megadását, az optimalizációs módszert beépített függvényekkel támogatják

Minta „sekély” hálózat TensorFlowban

```
X = tf.placeholder(tf.float32, [None, 10])
Y = tf.placeholder(tf.float32, [None, 1])
```

```
W1 = tf.Variable(tf.random_normal([10, 10]), name='W1')
b1 = tf.Variable(tf.random_normal([10]), name='b1')
W2 = tf.Variable(tf.random_normal([10, 8]), name='W2')
b2 = tf.Variable(tf.random_normal([8]), name='b2')
W3 = tf.Variable(tf.random_normal([8, 1]), name='W3')
b3 = tf.Variable(tf.random_normal([1]), name='b3')
```

```
Z1 = tf.add(tf.matmul(X, W1), b1)
A1 = tf.nn.relu(Z1)
Z2 = tf.add(tf.matmul(A1, W2), b2)
A2 = tf.nn.relu(Z2)
Z3 = tf.add(tf.matmul(A2, W3), b3)
```



Minta „sekély” hálózat TensorFlowban

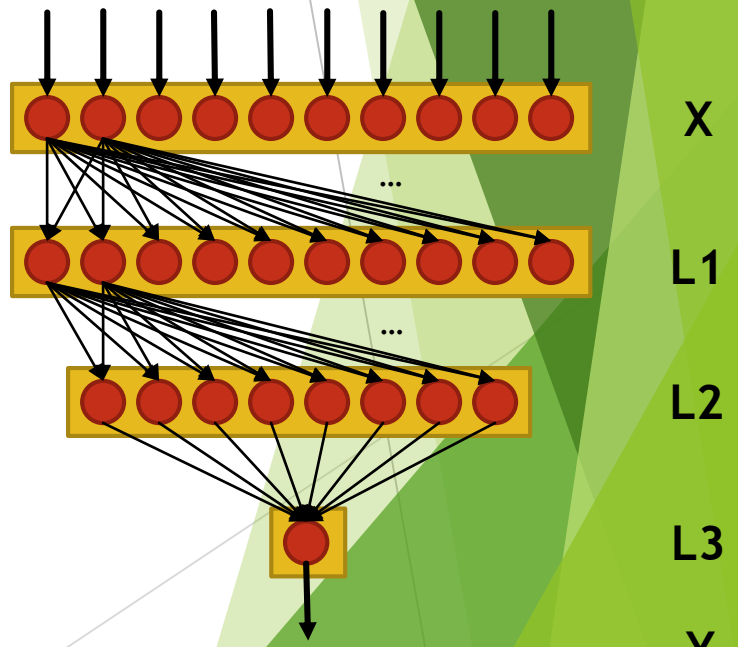
```
sigm = tf.nn.sigmoid(Z3, name='sigm')
loss = tf.reduce_mean(tf.nn.sigmoid_cross_entropy_with_logits(logits=Z3, labels=Y))
pred = tf.cast(tf.greater_equal(sigm, 0.5), tf.float32, name='pred')
acc = tf.reduce_mean(tf.cast(tf.equal(pred, Y), tf.float32), name='acc')
```

```
optimizer = tf.train.GradientDescentOptimizer(0.001).minimize(loss)
```

```
init = tf.global_variables_initializer()

with tf.Session() as sess:
    sess.run(init)

    for epoch in range(50):
        sess.run(optimizer, feed_dict={X: train_X, Y: train_Y})
```

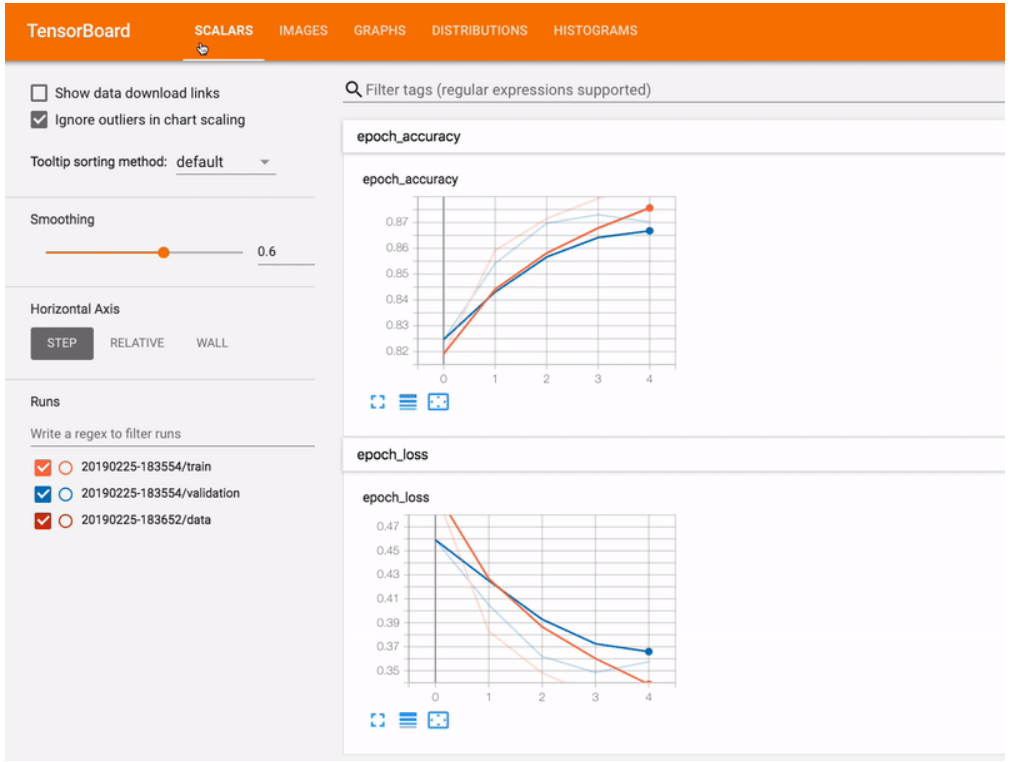


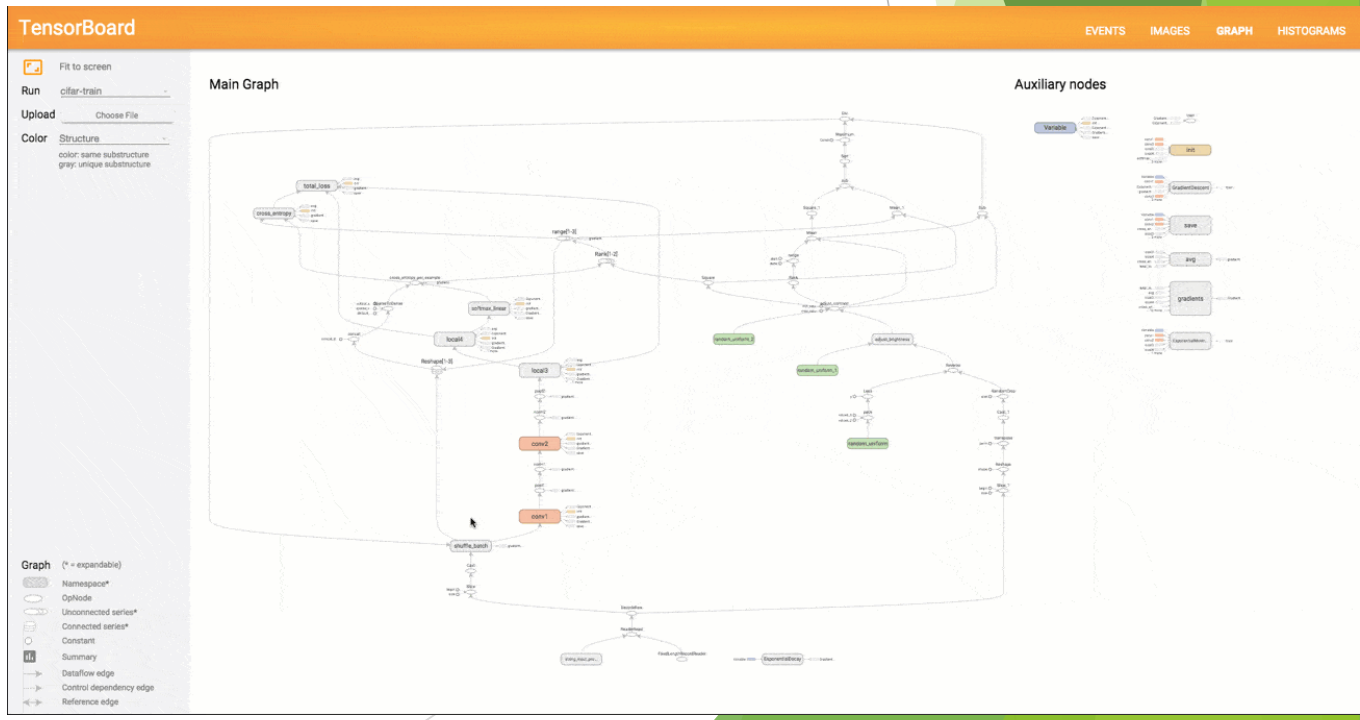
TensorFlow

- Ahogy gyorsan felfedezhető, a TensorFlow egy alacsony szintű APIt szolgáltat
- A szakértőnek alacsony szintű hozzáférést ad
- Ugyanakkor a gráfok definiálásakor nagy hibalehetőség
- A legtöbb hálózat építési blokkjai azonban megegyeznek, előre ismertek
  - Teljesen összekötött rétegek
  - Konvolúciós, Pooling rétegek
  - Rekurrens blokkok
  - stb

TensorBoard

- Egy gyakran használt eszköz a fejlesztés támogatására
- Webes felületen követhető a modell tanítása során a különböző paraméterek változása, de a modell architekturája is vizualizálható





Keras

- Felismerve, hogy a TensorFlow (és más frameworkok) jellemzően alacsony szintű hozzáférést adnak a fejlesztőknek, megjelent az igény a magas szintű APIkra

„Being able to go from idea to result with the least possible delay is key to doing good research.”

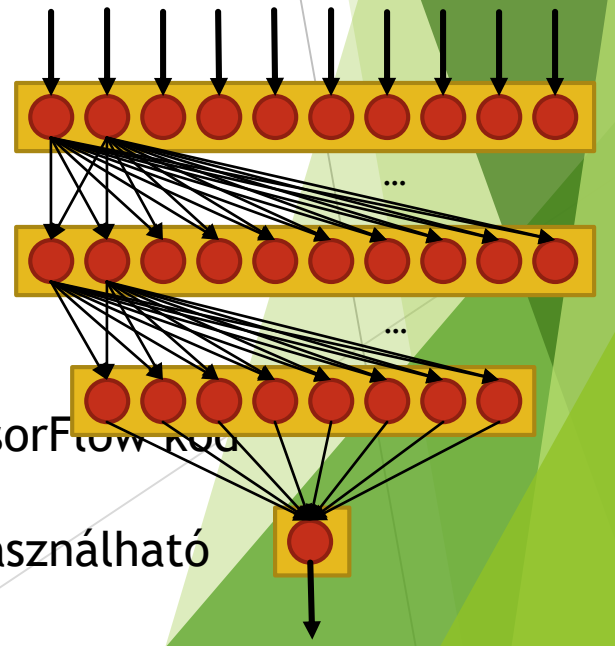
- Francois Chollet, a Google mérnöke 2015 óta fejleszti
- Felső rétegként használható több framework felett is, például TensorFlow is alkalmazható mint Keras backend

Keras

- Főbb tervezési szempontok a megtervezésekor
  - Felhasználóbarátság
  - Modularitás
  - Egyszerű bővíthetőség
  - Python
- A TensorFlow 2017 óta az alacsony szintű interfész mellett a Kerast hivatalosan támogatja, olyannyira, hogy a TF csomag részeként elérhető
  - A TensorFlow 2.0 (2019 szept) már alapértelmezetten Keras felülettel támogatott

Minta „sekély” hálózat Kerasban

```
model = Sequential()
model.add(Dense(10, activation='relu', input_shape=(10, )))
model.add(Dense(8, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
model.compile(
      optimizer=tf.train.GradientDescentOptimizer(learning_rate=0.0001),
      loss='binary_crossentropy', metrics=['accuracy'])
model.fit(train_X, train_Y,
      epochs=50)
```

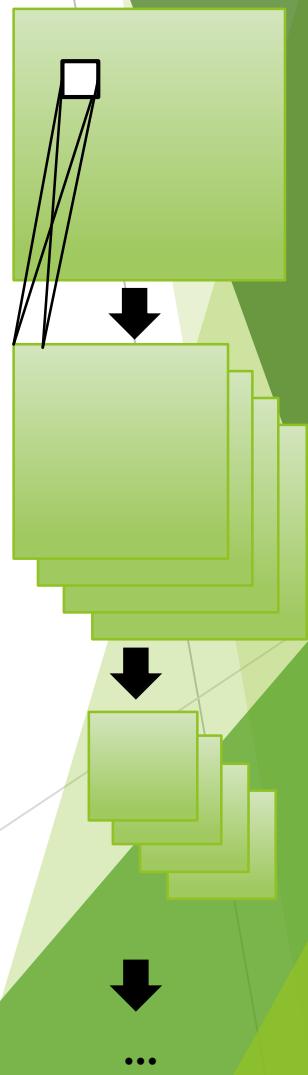


- Ez ugyanazt a feladatot hajtja végre, mint a korábban bemutatott TensorFlow kód
- A leírás sokkal rövidebb, olvashatóbb, a hibalehetőség kisebb
- A pozitív oldal megmaradt: a TF függvényei elérhetőek, TensorBoard használható

Konvolúciós hálózat

```
model = Sequential()
model.add(Convolution2D(32, (7, 7),
                        input_shape=(128, 128, 3),
                        activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Convolution2D(64, (5, 5),activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Convolution2D(128, (3, 3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Convolution2D(256, (3, 3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(400, activation='relu'))
model.add(Dense(1, activation='sigmoid'))

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



Köszönöm a figyelmet!

Köszönöm a figyelmet!

12:00 – 13:00

Ebédszünet

13:00 - 14:25

Keras és TensorFlow referencia architektúrák indítása és használata az ELKH Cloudon (Farkas Attila, SZTAKI)

14:25 - 14:40

Kávészünet

14:40 - 16:00

Esettanulmány: egy komplex képi klasszifikációs feladat hatékony megoldása felhő környezetben (Kertész Gábor, SZTAKI)

Kérdések és válaszok (Kacsuk Péter, Kertész Gábor, Farkas Attila)