



Keras és TensorFlow referencia architektúrák az ELKH Cloudon



Farkas Attila
farkas.attila@sztaki.hu

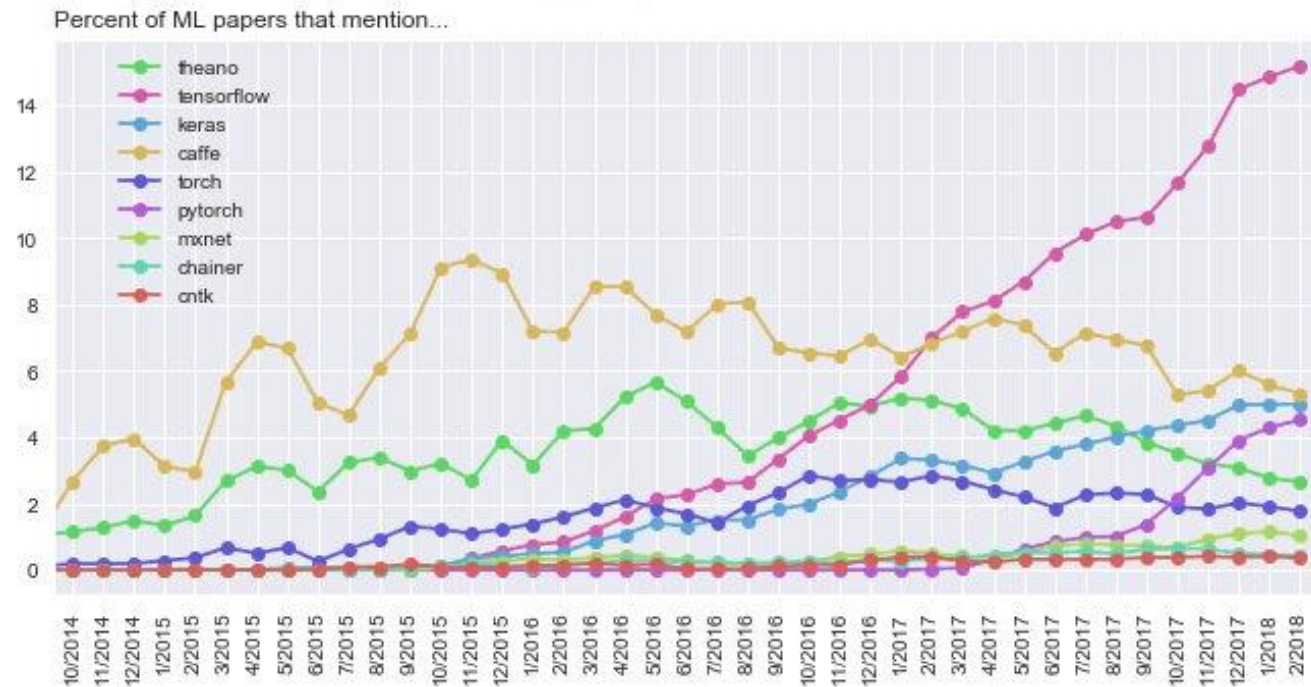


ELKH Cloud

TensorFlow és Keras

TensorFlow

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



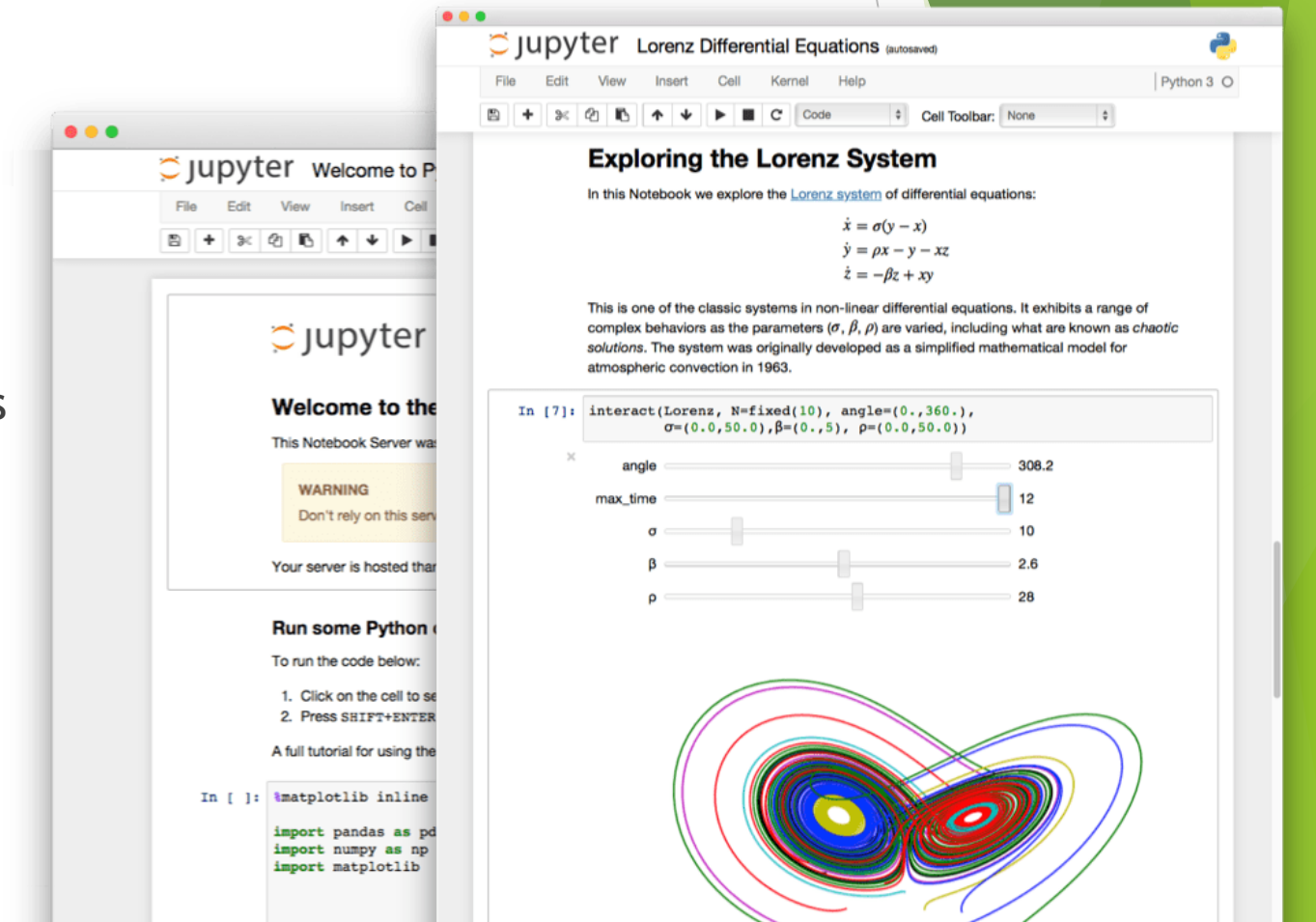
Keras



- ▶ 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

Jupyter Notebook

- ▶ Nyílt forráskódú webalkalmazás
- ▶ Fejlesztő környezet biztosít
- ▶ Adat vizualizációs megoldás
- ▶ Széleskörű programozási nyelv támogatás
- ▶ A Notebookok könnyedén megoszthatók



jupyter Lorenz Differential Equations (autosaved)

File Edit View Insert Cell Kernel Help Python 3

Exploring the Lorenz System

In this Notebook we explore the [Lorenz system](#) of differential equations:

$$\begin{aligned}\dot{x} &= \sigma(y - x) \\ \dot{y} &= \rho x - y - xz \\ \dot{z} &= -\beta z + xy\end{aligned}$$

This is one of the classic systems in non-linear differential equations. It exhibits a range of complex behaviors as the parameters (σ , β , ρ) are varied, including what are known as *chaotic solutions*. The system was originally developed as a simplified mathematical model for atmospheric convection in 1963.

```
In [7]: interact(Lorenz, N=fixed(10), angle=(0.,360.),
                sigma=(0.0,50.0), beta=(0.,5), rho=(0.0,50.0))
```

angle 308.2
max_time 12
 σ 10
 β 2.6
 ρ 28

```
In [ ]: %matplotlib inline
import pandas as pd
import numpy as np
import matplotlib
```

JupyterLab



- ▶ Jupyter Notebook továbbfejlesztett verziója
- ▶ Web alapú interaktív fejlesztőkörnyezet
- ▶ Terminal biztosítása
- ▶ Moduláris felépítés
- ▶ Bővítmények támogatása

A screenshot of the JupyterLab web interface. The main window displays a notebook titled "In Depth: Linear Regression" with text explaining regression models. Below the text is a code cell with a plot. To the right, there's a "Launcher" panel with icons for Python 3, C++11, C++14, C++17, Julia 1.1.0, phylogenetics (Python 3.7), and R. Further right is an "Output View" showing a scatter plot of "Seattle Weather: 2012-2015" with a bar chart below it. At the bottom, there are three smaller notebook windows: "Julia", "python notebook", and "R". The "python notebook" window shows code for solving the Lorenz system. The "R" window shows a scatter plot of the Iris dataset. A terminal window is visible at the bottom left, showing code execution output.

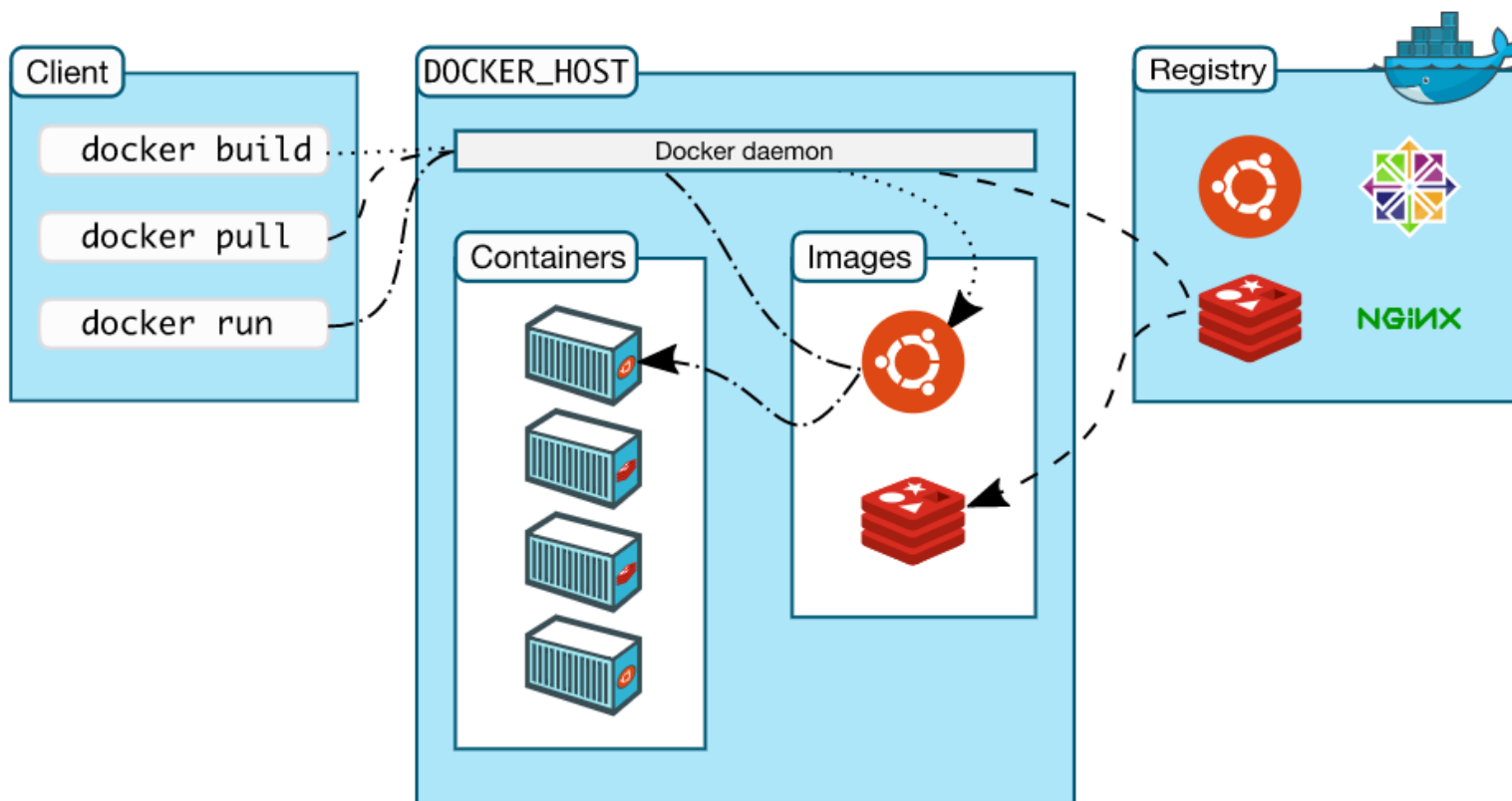


Docker

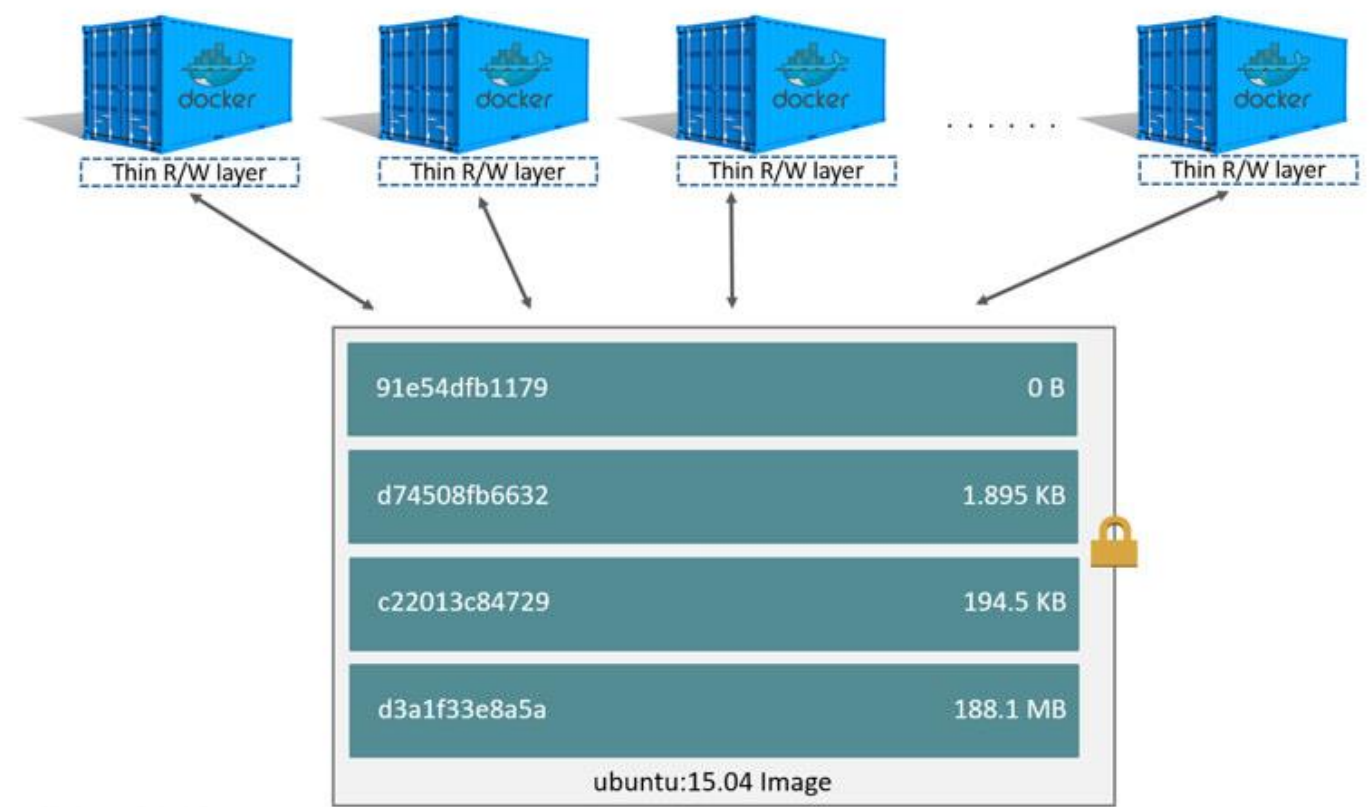
Docker konténer technológia

- ▶ Nyílt konténer platform
- ▶ Virtualizáció helyettesítése
- ▶ Szeparált alkalmazás futtatás
- ▶ Linux konténer technológián alapszik
- ▶ Réteges felépítésű konténerek
- ▶ Központosított képfájl tárolás
- ▶ Hordozhatóság

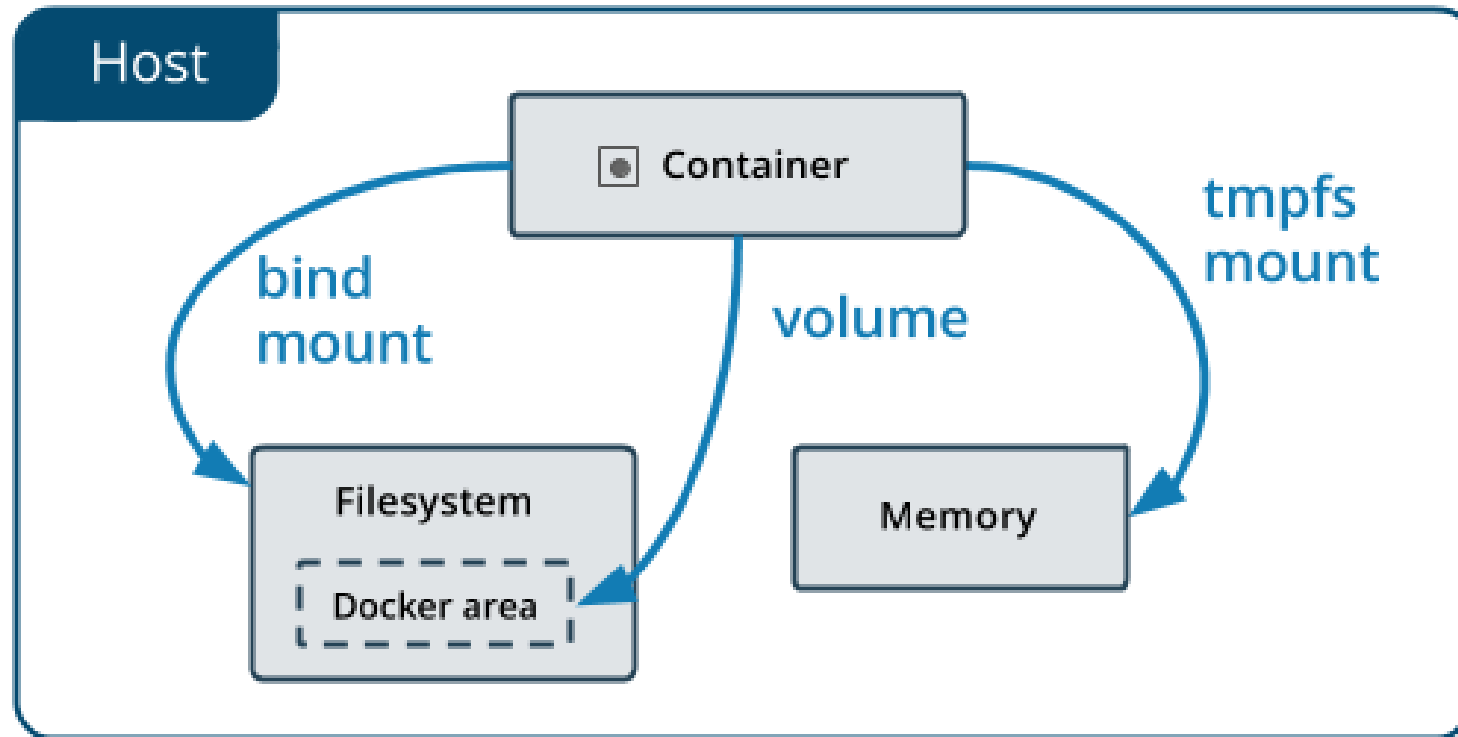
Docker architektúra



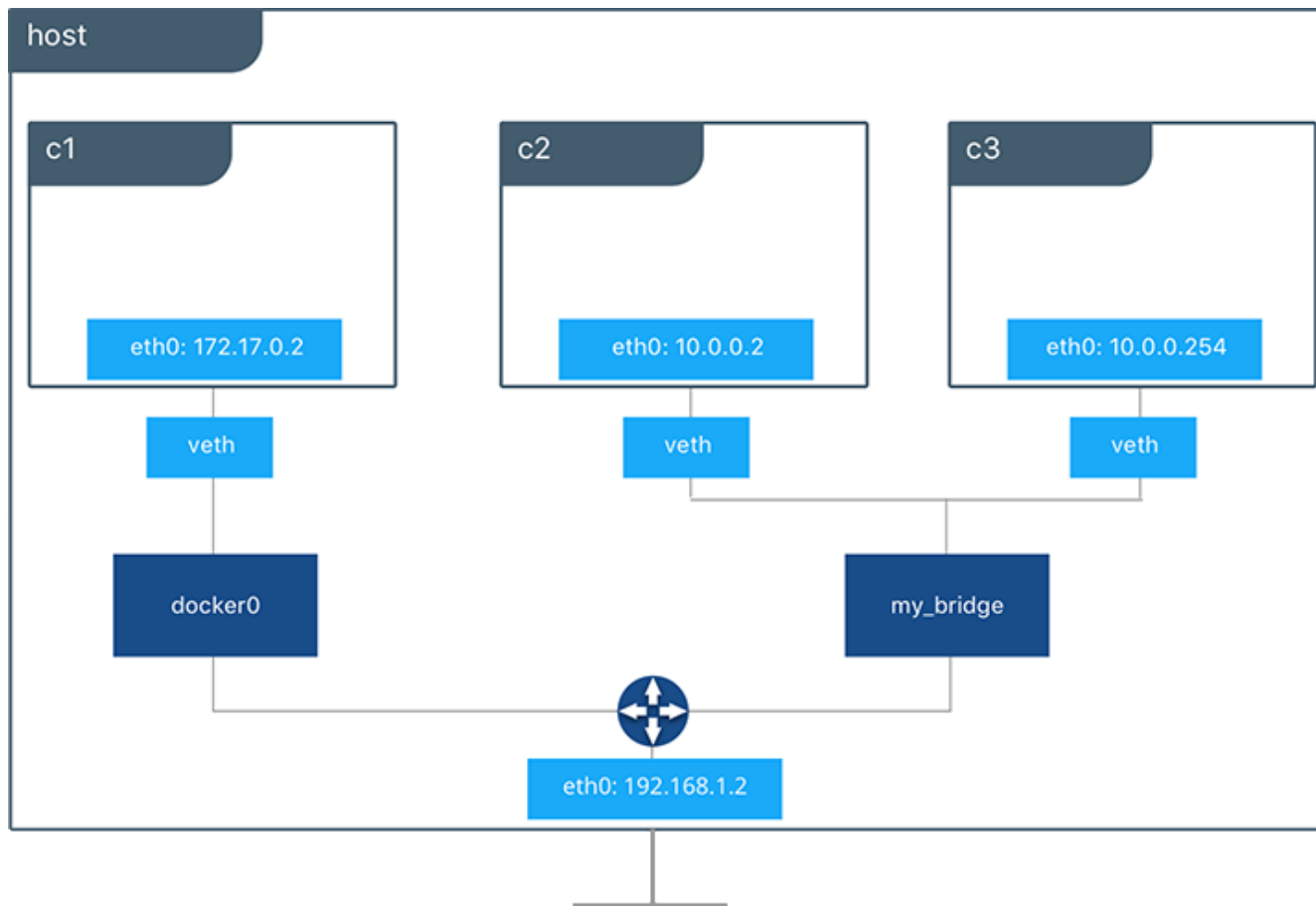
Docker konténer



Docker volume



Docker virtuális hálózat

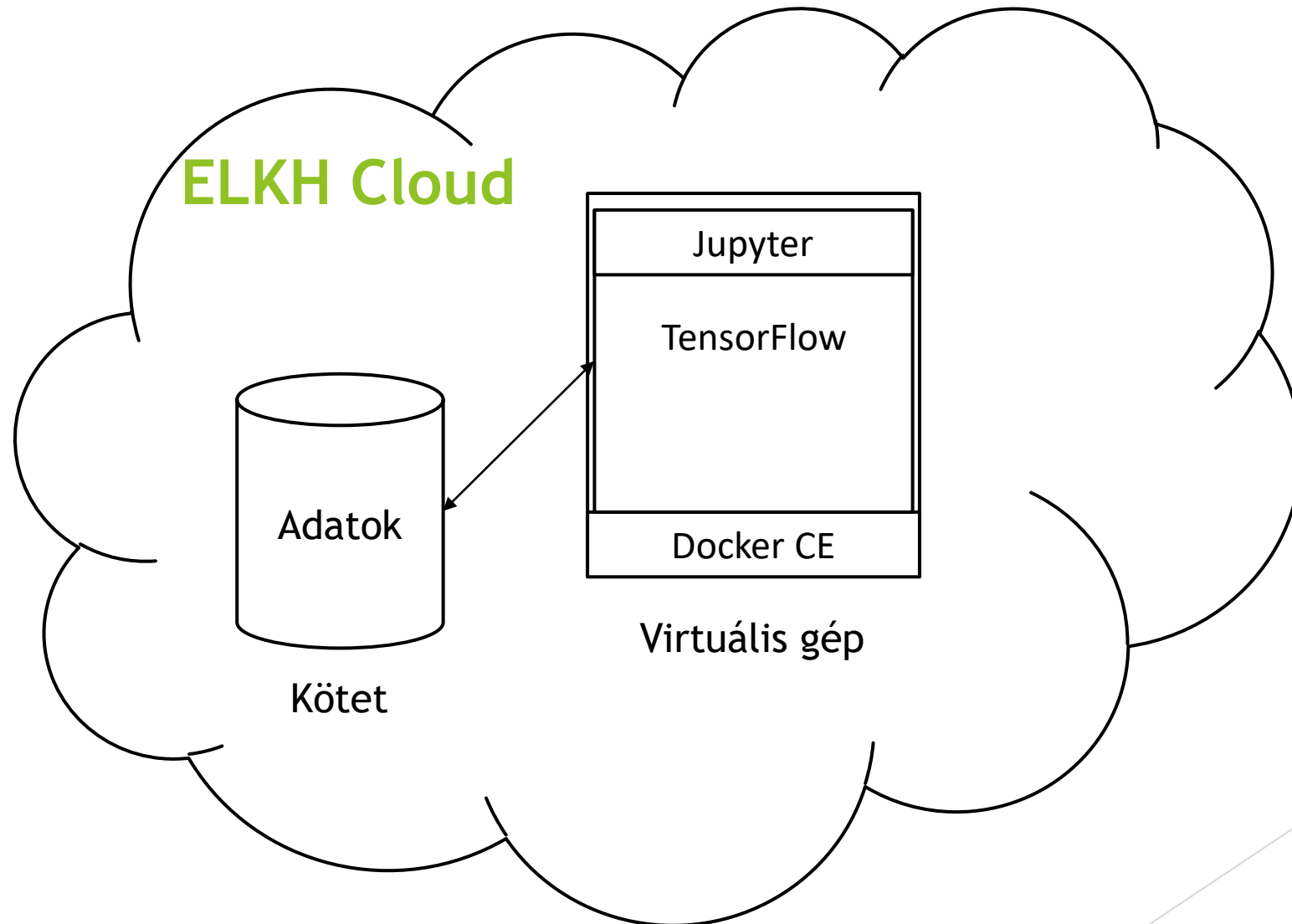




ELKH Cloud

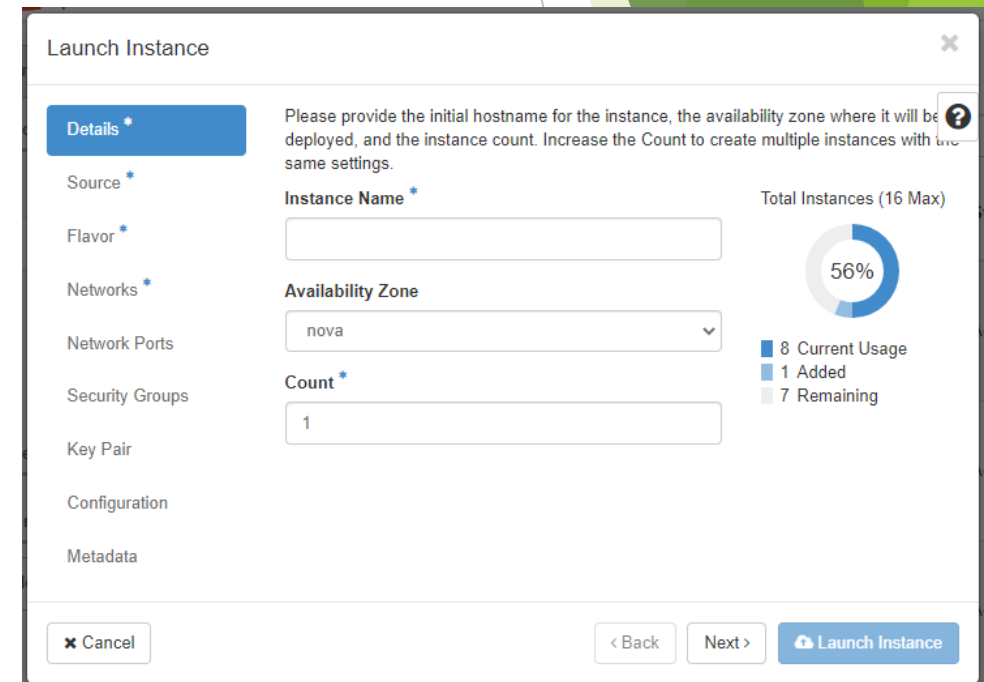
TensorFlow használata Docker keretrendszerben ELKH Cloudon

Architektúra



Linux virtuális gép létrehozása

- ▶ Távoli elérés biztosítása
 - ▶ SSH kulcs generálása vagy meglévő kulcs feltöltése
 - ▶ Biztonsági csoport létrehozása, a távoli elérés biztosításához
 - ▶ TCP 22 és 8888 portok szükségesek
- ▶ Virtuális gép létrehozása (részletek a Bevezető oktatásban)
 - ▶ Ubuntu 16.04 vagy újabb forrás képfájl felhasználása
 - ▶ Minimum m1.small méret kiválasztása
 - ▶ Megfelelő biztonsági csoport kiválasztása
 - ▶ Generált vagy feltöltött kulcs hozzáadása
- ▶ Kötet hozzárendelése
- ▶ Távoli eléréshez Floating IP hozzárendelése



Launch Instance

Details *

Source *

Flavor *

Networks *

Network Ports

Security Groups

Key Pair

Configuration

Metadata

Please provide the initial hostname for the instance, the availability zone where it will be deployed, and the instance count. Increase the Count to create multiple instances with the same settings.

Instance Name *

Availability Zone

Count *

Total Instances (16 Max)

56%

8 Current Usage
1 Added
7 Remaining

Cancel

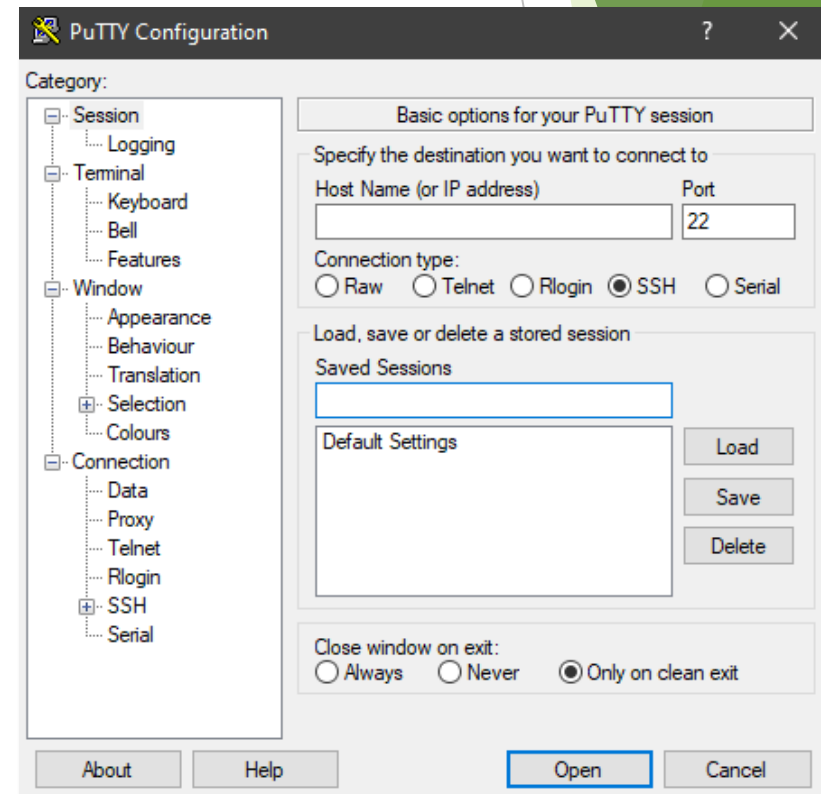
< Back

Next >

Launch Instance

Csatlakozás a létrehozott virtuális géphez

- ▶ Töltsük le és telepítsük a PuTTY programot:
 - ▶ <https://www.putty.org>
- ▶ A programot elindítva adjuk meg a következőket:
 - ▶ **Host Name:** a virtuális gép külső IP címe
 - ▶ **Port:** a géphez kapcsolódó SSH port, alapból 22-es
- ▶ A generált és letöltött vagy már meglévő privát kulcs felhasználása, amelynek publikus párját hozzáadtuk a virtuális géphez
- ▶ Csatlakozás a virtuális géphez



Docker CE telepítés a létrehozott virtuális gépen

► Előfeltételek telepítése

```
$ sudo apt-get update  
$ sudo apt-get install \  
  apt-transport-https \  
  ca-certificates \  
  curl \  
  gnupg-agent \  
  software-properties-common
```

► Docker repository hozzáadása

```
$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -  
$ sudo add-apt-repository \  
  "deb [arch=amd64] https://download.docker.com/linux/ubuntu \  
  $(lsb_release -cs) \  
  stable"
```

Docker CE telepítés #2

▶ Docker CE telepítése

```
$ sudo apt-get update  
$ sudo apt-get install docker-ce docker-ce-cli containerd.io
```

▶ Docker futtatása root jogosultság nélkül

```
$ sudo groupadd docker  
$ sudo usermod -aG docker $USER
```

▶ Újra bejelentkezés után érvénybe lépnek a csoport jogosultságok

```
$ docker run hello-world
```

Docker parancs felépítése

```
$ docker run -d -p 8080:80 --rm --name app --network mynet python-app:latest
```

Docker parancs
vagy „objektum”

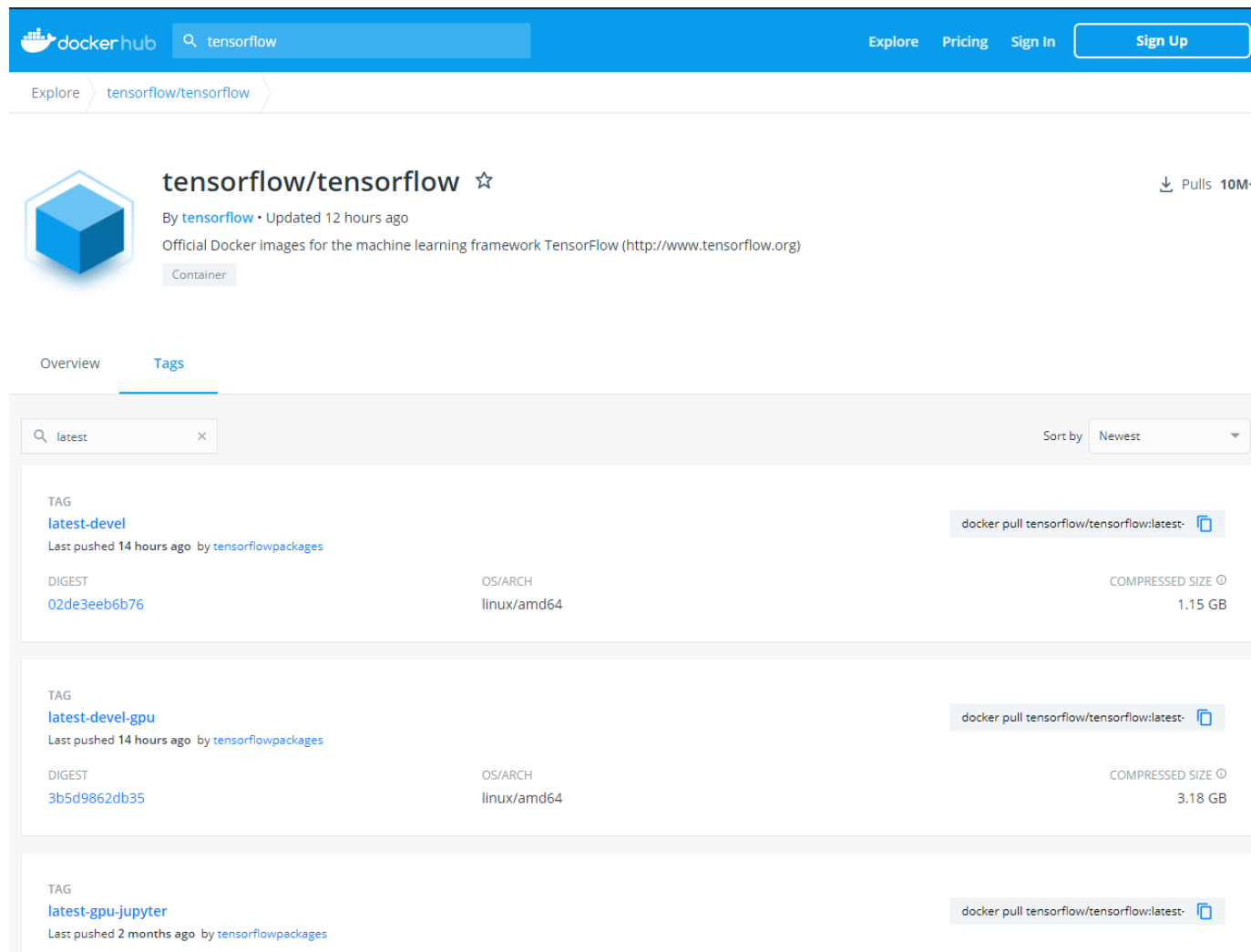
Kapcsolók értékekkel
(opcionális)

Bemeneti attribútum

Alapparancs

- ▶ A példa parancs a bementi attribútumként megadott konténert fogja elindítani, a megadott kapcsolóknak megfelelően
- ▶ A parancsoknak minden esetben a „docker” kulcsszóval kell kezdődnie
- ▶ Ezt követően meg kell határozni, hogy mely docker parancsot vagy mely docker „objektummal” szeretnék parancsot végrehajtani
 - ▶ Docker parancs: build, pull, push, run, ls, rm, stb.
 - ▶ Docker „objektum” pl.: container, image, network, volume, stb.
 - ▶ Minden objektumnak megvannak a saját futtatható parancsai
 - ▶ Pl.: docker container run

Jupyter Docker képfájlok



The screenshot shows the Docker Hub page for the tensorflow/tensorflow repository. The page is titled "tensorflow/tensorflow" and is marked as a "Container". It features a search bar with "tensorflow" entered, navigation links for "Explore", "Pricing", "Sign In", and "Sign Up", and a "Pulls 10M+" indicator. The main content area is divided into "Overview" and "Tags" tabs. The "Tags" tab is active, displaying a list of tags with their respective details. The tags listed are "latest-devel", "latest-devel-gpu", and "latest-gpu-jupyter". Each tag entry includes the tag name, the time it was last pushed, the OS/ARCH, the compressed size, and a "docker pull" command.

TAG	OS/ARCH	COMPRESSED SIZE
latest-devel Last pushed 14 hours ago by tensorflowpackages	linux/amd64	1.15 GB
latest-devel-gpu Last pushed 14 hours ago by tensorflowpackages	linux/amd64	3.18 GB
latest-gpu-jupyter Last pushed 2 months ago by tensorflowpackages		

<https://hub.docker.com/r/tensorflow/tensorflow>

Jupyter konténer elindítása

```
$ docker run -p 8888:8888 -v /mnt/data:/tf/notebooks --name jupyter tensorflow/tensorflow:latest-jupyter
```

```
ubuntu@fattila-gpu:~$ sudo docker run -p 8888:8888 --name jupyter tensorflow/tensorflow:latest-jupyter
[I 19:54:09.690 NotebookApp] Writing notebook server cookie secret to /root/.local/share/jupyter/runtime/notebook_cookie_secret
[I 19:54:10.011 NotebookApp] Serving notebooks from local directory: /tf
[I 19:54:10.011 NotebookApp] Jupyter Notebook 6.1.4 is running at:
[I 19:54:10.011 NotebookApp] http://f65d5408b41e:8888/?token=ea4241d5bf033e83f93979ff38fa04ed62c8e21963813271
[I 19:54:10.011 NotebookApp] or http://127.0.0.1:8888/?token=ea4241d5bf033e83f93979ff38fa04ed62c8e21963813271
[I 19:54:10.011 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 19:54:10.016 NotebookApp]
```

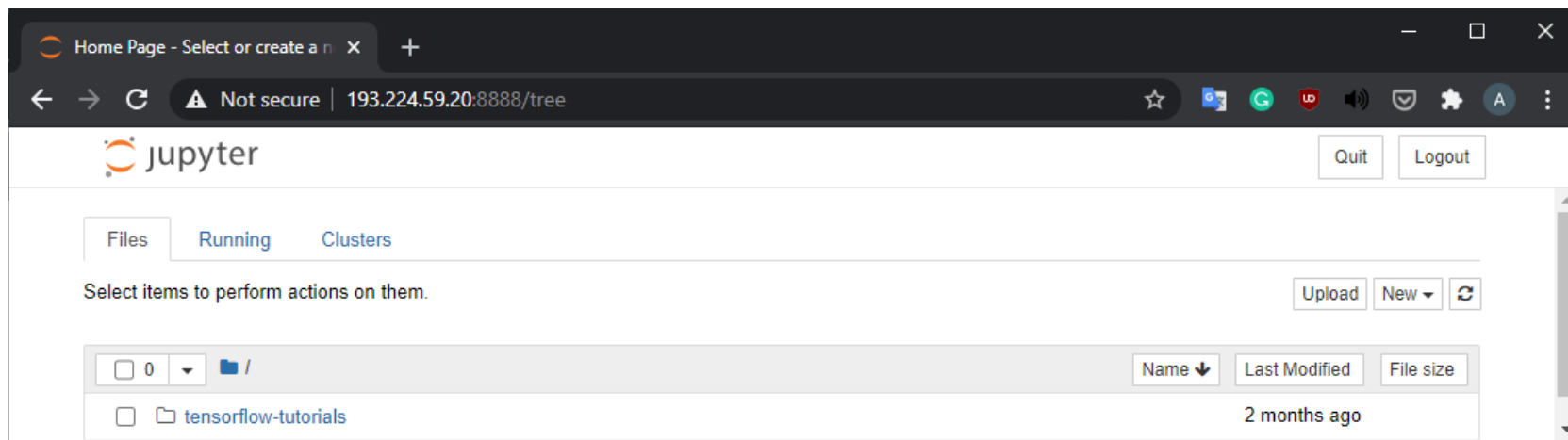
To access the notebook, open this file in a browser:

file:///root/.local/share/jupyter/runtime/nbserver-1-open.html

Or copy and paste one of these URLs:

http://f65d5408b41e:8888/?token=ea4241d5bf033e83f93979ff38fa04ed62c8e21963813271

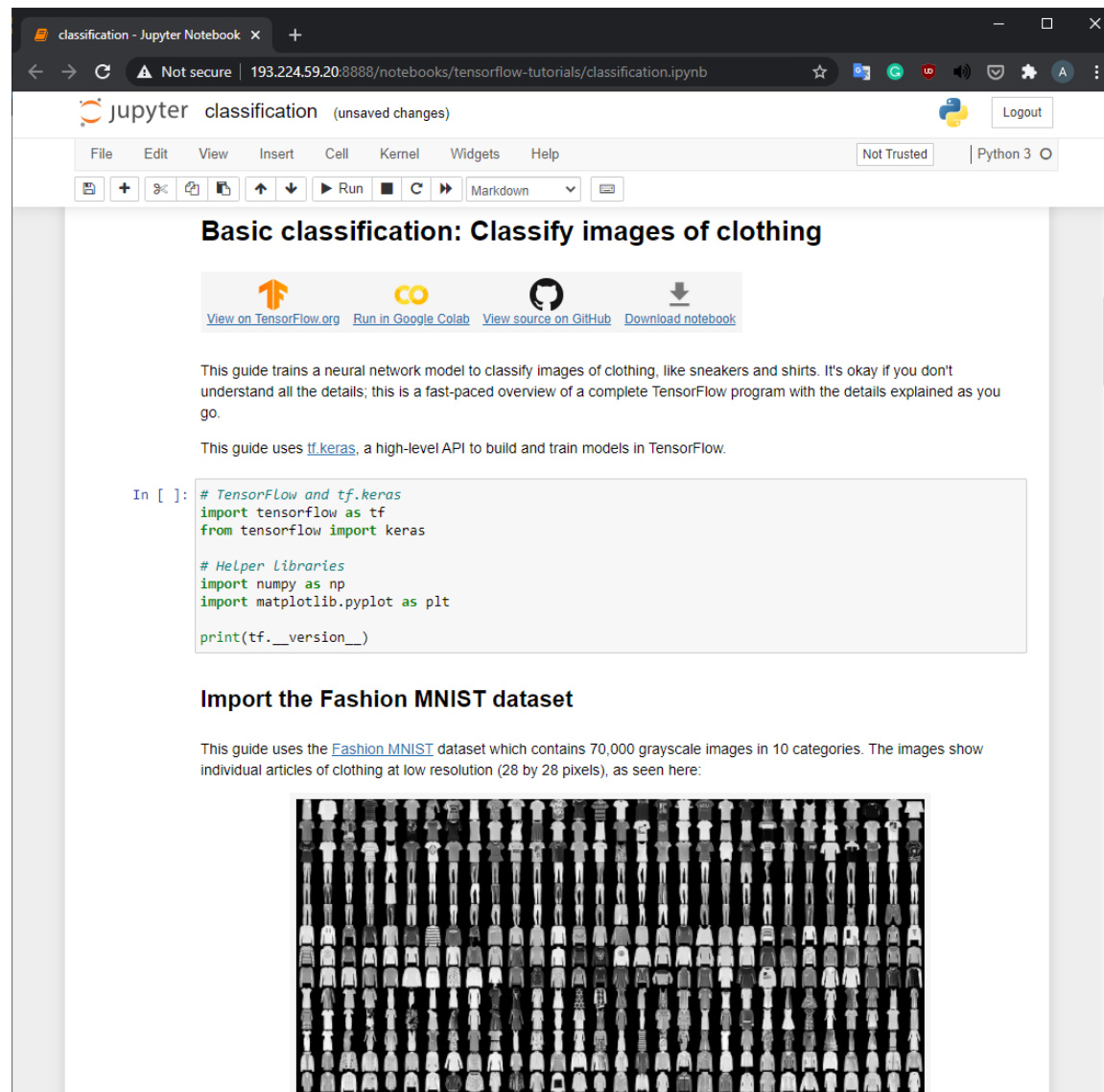
or http://127.0.0.1:8888/?token=ea4241d5bf033e83f93979ff38fa04ed62c8e21963813271



```
In [1]: import tensorflow as tf
print(tf.__version__)

2.3.1
```

TensorFlow használata Jupyter környezetben



The screenshot displays a Jupyter Notebook titled "classification - Jupyter Notebook" in a web browser. The browser address bar shows the URL "193.224.59.20:8888/notebooks/tensorflow-tutorials/classification.ipynb". The notebook interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help), a toolbar with icons for file operations and execution, and a status bar indicating "Not Trusted" and "Python 3".

Basic classification: Classify images of clothing

[View on TensorFlow.org](#) [Run in Google Colab](#) [View source on GitHub](#) [Download notebook](#)

This guide trains a neural network model to classify images of clothing, like sneakers and shirts. It's okay if you don't understand all the details; this is a fast-paced overview of a complete TensorFlow program with the details explained as you go.

This guide uses [tf.keras](#), a high-level API to build and train models in TensorFlow.


```
In [ ]: # TensorFlow and tf.keras
import tensorflow as tf
from tensorflow import keras

# Helper Libraries
import numpy as np
import matplotlib.pyplot as plt

print(tf.__version__)
```

Import the Fashion MNIST dataset

This guide uses the [Fashion MNIST](#) dataset which contains 70,000 grayscale images in 10 categories. The images show individual articles of clothing at low resolution (28 by 28 pixels), as seen here:

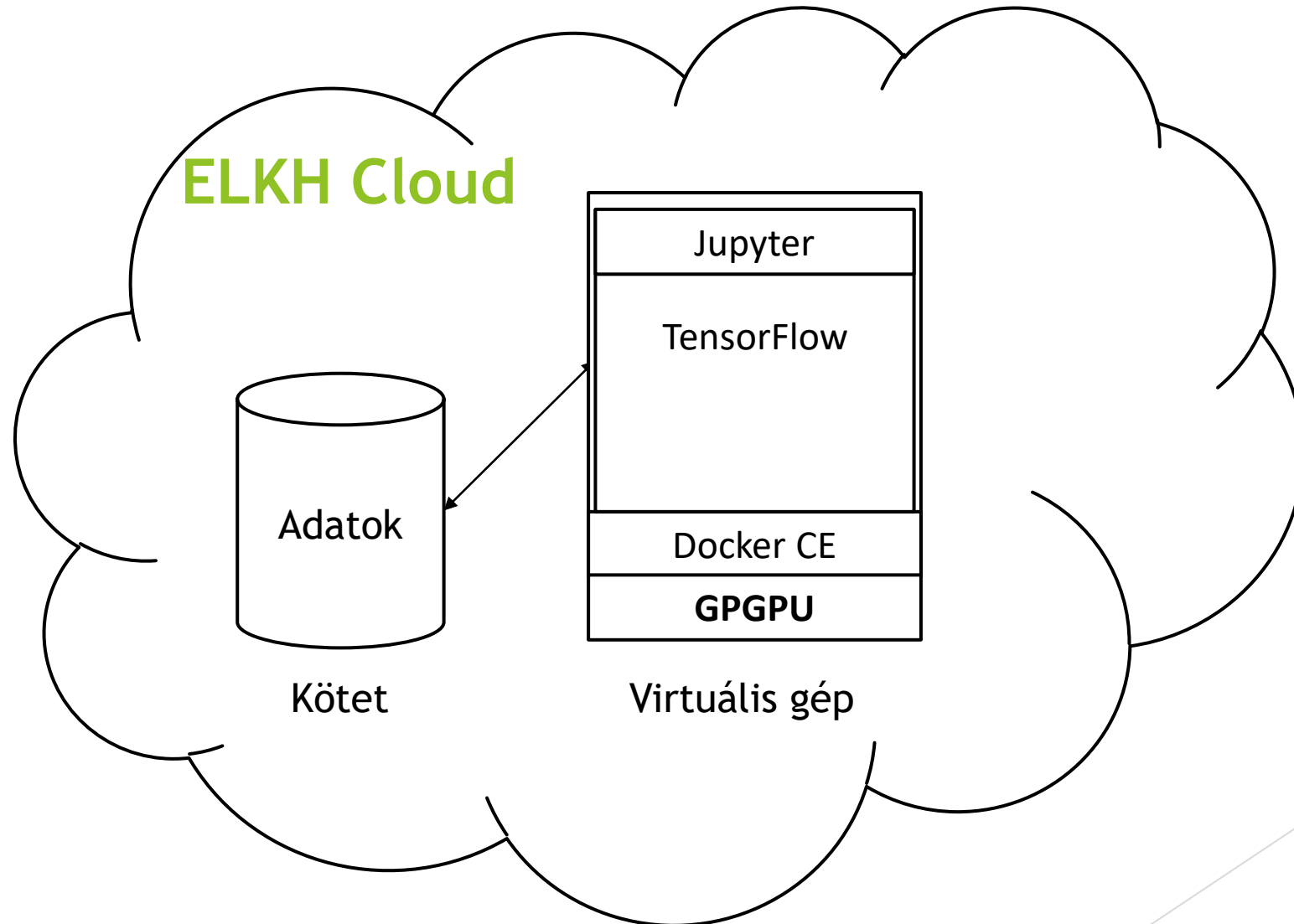




ELKH Cloud

TensorFlow használata ELKH Cloudon GPU erőforrásokon

Architektúra GPGPU-val



Linux virtuális gép létrehozása GPU erőforrással

- ▶ Linux virtuális gép létrehozása, a korábbi példa alapján
 - ▶ `oktatas.k80` flavour használata, ami tartalmaz egy NVIDIA Tesla K80-as videokártyát
- ▶ NVIDIA driver telepítése (részletek a Bevezető oktatásban)
- ▶ Docker CE telepítése, a korábbi példa alapján
- ▶ NVIDIA Container Toolkit telepítése:

```
$ distribution=$(. /etc/os-release;echo $ID$VERSION_ID) \  
  && curl -s -L https://nvidia.github.io/nvidia-docker/gpgkey | sudo apt-key add - \  
  && curl -s -L https://nvidia.github.io/nvidia-docker/$distribution/nvidia-docker.list | \  
sudo tee /etc/apt/sources.list.d/nvidia-docker.list  
  
$ sudo apt-get update  
$ sudo apt-get install -y nvidia-docker2  
$ sudo systemctl restart docker
```

GPU erőforrások ellenőrzése

```
$ docker run --rm --gpus all nvidia/cuda:11.0-base nvidia-smi
```

```
ubuntu@fattila-gpu:~$ sudo docker run --rm --gpus all nvidia/cuda:11.0-base nvidia-smi
Unable to find image 'nvidia/cuda:11.0-base' locally
11.0-base: Pulling from nvidia/cuda
54ee1f796a1e: Pull complete
f7bfea53ad12: Pull complete
46d371e02073: Pull complete
b66c17bbf772: Pull complete
3642f1a6dfb3: Pull complete
e5ce55b8b4b9: Pull complete
155bc0332b0a: Pull complete
Digest: sha256:774ca3d612de15213102c2dbbba55df44dc5cf9870ca2be6c6e9c627fa63d67a
Status: Downloaded newer image for nvidia/cuda:11.0-base
Tue Nov 24 19:35:47 2020

+-----+
| NVIDIA-SMI 455.45.01   Driver Version: 455.45.01   CUDA Version: 11.1   |
+-----+
| GPU Name          Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|      Memory-Usage | GPU-Util  Compute M. |
|                               |              MIG M. |
+-----+
| 0   Tesla K80           On          | 00000000:00:05.0 Off |             0         |
| N/A   41C    P8      30W / 149W |      0MiB / 11441MiB |           0%      Default |
|                               |                      |              N/A         |
+-----+

+-----+
| Processes:
| GPU  GI  CI           PID  Type  Process name                        GPU Memory
|   ID  ID                                     Usage
+-----+
| No running processes found
+-----+
```

TensorFlow használata Jupyter környezetben GPU erőforrással

```
$ docker run --gpus all -p 8888:8888 --name jupyter -v /mnt/data:/tf/notebooks \  
tensorflow/tensorflow:latest-gpu-jupyter
```

```
In [1]: import tensorflow as tf  
print("Num GPUs Available: ", len(tf.config.experimental.list_physical_devices('GPU')))
```

```
Num GPUs Available: 1
```

```
In [2]: tf.config.experimental.list_physical_devices('GPU')
```

```
Out[2]: [PhysicalDevice(name='/physical_device:GPU:0', device_type='GPU')]
```



ELKH Cloud

Köszönöm a figyelmet!