Lielapjoma pilsētvides video analīze, izmantojot Dziļo apmācību un HPC resursus

K.Sudars, R.Kadiķis, R.Cacurs no BigData grupas

Aktivitāte: Lielapjoma pilsētvides video un citu sensoro signālu savākšana un analīze, izmantojot augstas veiktspējas skaitļošanas resursus

Uzdevums: Attīstīt video analīzes algoritmus un metodes iepriekš zināmu darbību vai notikumu scenāriju atpazīšanai pilsētas drošības veicināšanai

1. posma apakšuzdevumi:
* Apzināt labākos tehnoloģiskos risinājumus video analīzei (Rezultāts: MNT)
* Iekārtot vidi MNT apmācībai
(Rezultāts: Serveris ar 4 x Nvidia Tesla K20 GPU + *Caffe*, *Torch*)

2. posma apakšuzdevumi:
* Divu klašu apmācības piemērs
* Skaitļošanas vides un rīku izvēle un sakārtošana
(Rezultāts: *TensorFlow* izvēlēta kā galvenā MNT apmācības vide)
* MNT arhitektūra videoanalīzes uzdevuma risināšanai
(Rezultāts: Dati → CNN →RNN → Rezultāts)

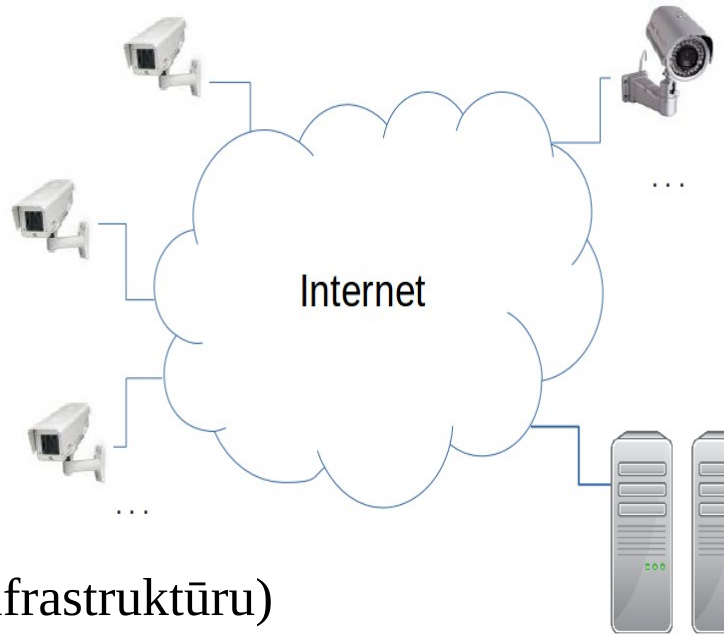
Grupas cilvēki: K.Sudars, R.Kadiķis, R.Cacurs, V.Plociņš, K.Ozols

2. posma darbības virzieni:

* Skaitļošanas vides sakārtošana
  Serveris ar 4 x Nvidia Tesla K20 GPU => Klāsteris ar 12 K40 GPU + Linux Rocks

* Dziļās apmācības rīku analīze
  Pārslēgšanās darbā uz TF (iznācis 2015 nov.)

* MNT arhitektūra uzdevuma risināšanai
  "Dati → CNN →RNN → Rezultāts"
  Face2vec, Autoencoder, Objektu skaitīšana



Data gathering and analysis servers

Datu savākšanas un apstrādes koncepcija:

* Izmantot esošo infrastruktūru
(gan eksistējošus hw/sw risinājumus, gan Edi infrastruktūru)

* Edi ***TestBed*** video savākšana ar 40 videokamerām
(simulē reālos datu savākšanas apstākļus)

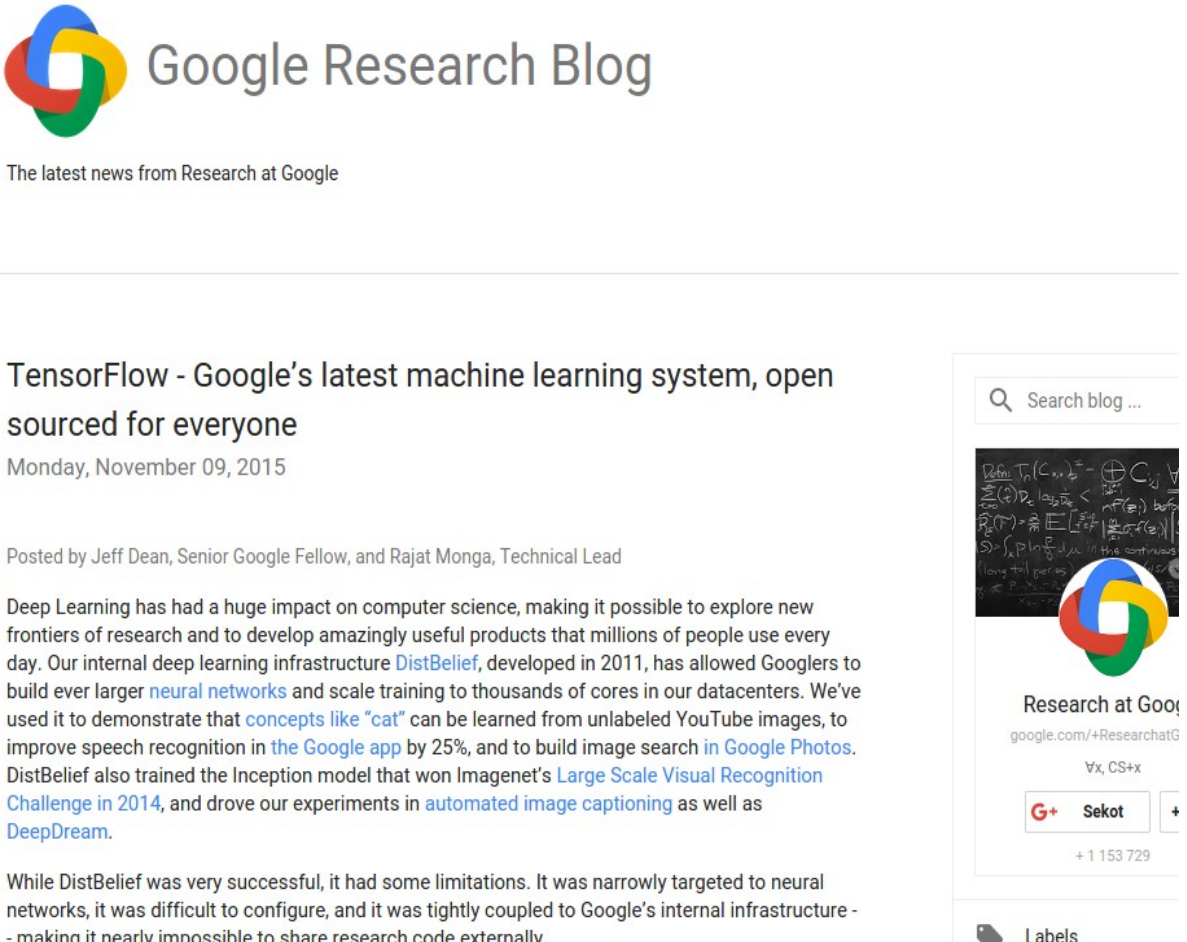
* Savākto datu klāsterizācija ar *Dziļo apmācību* konkrētu problēmu risināšanai
(piem., objektu atpazīšana dažādās kamerās, sekošana objektiem, personu identificējošas informācijas slēpšana video tml.)

TensorFlow - Google’s latest machine learning system, open sourced for everyone

http://googleresearch.blogspot.com/2015/11/tensorflow-googles-latest-machine_9.html

Priekšrocības:
- Python API
- multi GPU
- fleksibilitāte (p., LSTM, loops)
- Google komandas atbalsts
- Autograd

Alternatīvas:
- Caffe
- Torch
- Veles no Samsung
- Digits u.c.



Google Research Blog

The latest news from Research at Google

TensorFlow - Google’s latest machine learning system, open sourced for everyone

Monday, November 09, 2015

Posted by Jeff Dean, Senior Google Fellow, and Rajat Monga, Technical Lead

Deep Learning has had a huge impact on computer science, making it possible to explore new frontiers of research and to develop amazingly useful products that millions of people use every day. Our internal deep learning infrastructure DistBelief, developed in 2011, has allowed Googlers to build ever larger neural networks and scale training to thousands of cores in our datacenters. We've used it to demonstrate that concepts like “cat” can be learned from unlabeled YouTube images, to improve speech recognition in the Google app by 25%, and to build image search in Google Photos. DistBelief also trained the Inception model that won Imagenet's Large Scale Visual Recognition Challenge in 2014, and drove our experiments in automated image captioning as well as DeepDream.

While DistBelief was very successful, it had some limitations. It was narrowly targeted to neural networks, it was difficult to configure, and it was tightly coupled to Google's internal infrastructure - - making it nearly impossible to share research code externally

Search blog ...

Research at Goog

google.com/+ResearchatG

∀x, CS+x

Sekot

+ 1 153 729

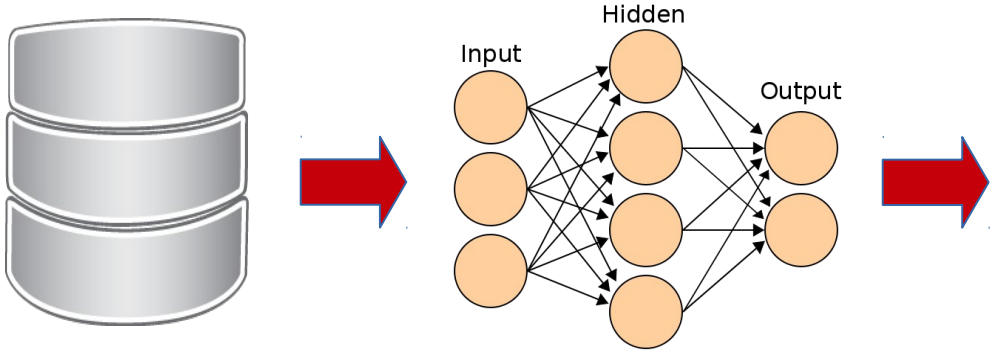
Labels

“Kaķis vai Zaķis” detektors DIGITS-2 vidē

Klasifikācija pie marķētiem datiem

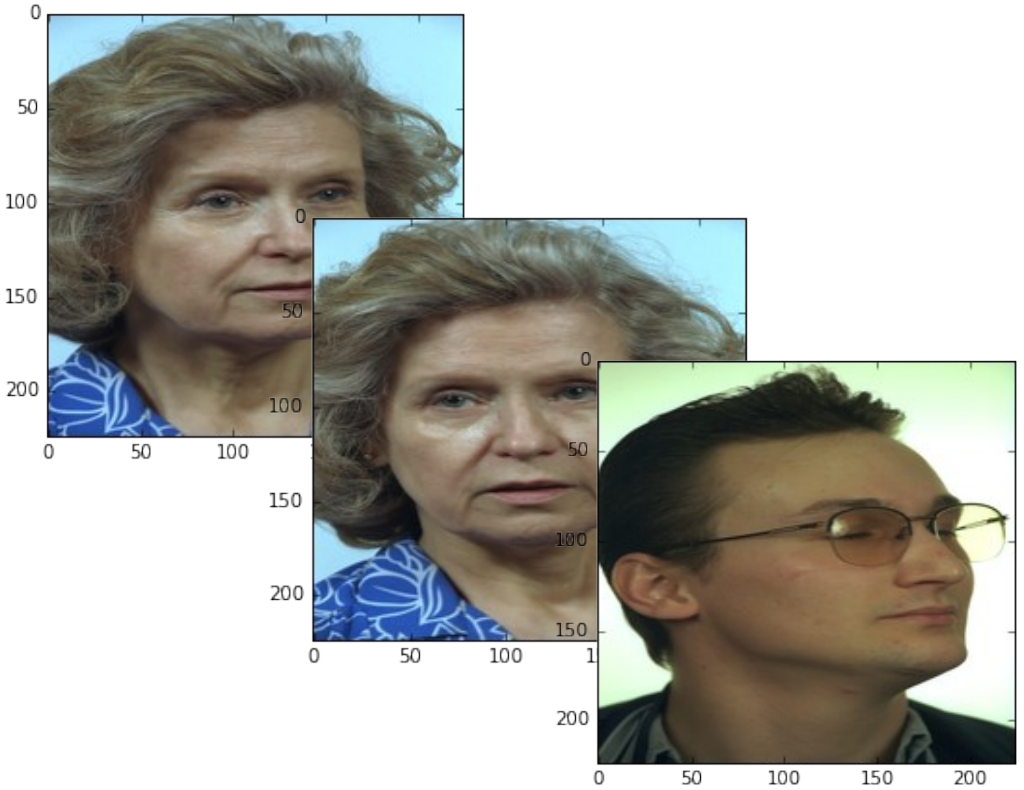
Face2vec
Unikālu objektu atpazīšana

COLOR-FERET
Treniņa kopa:
10148 attēli
905 klases
Pārbaudes kopa:
1190 bildes
108 klases



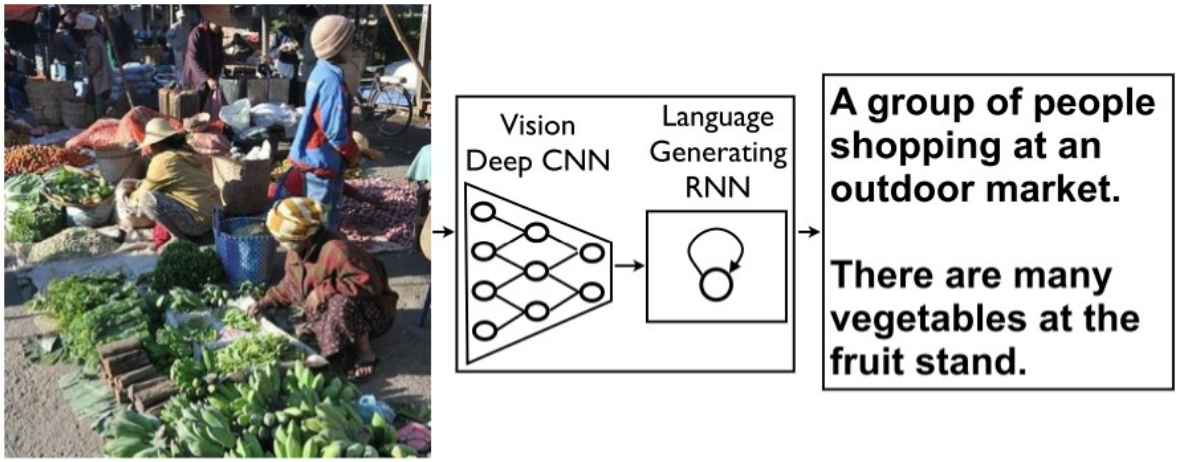
AlexNet ANN

Pazīmju vektors

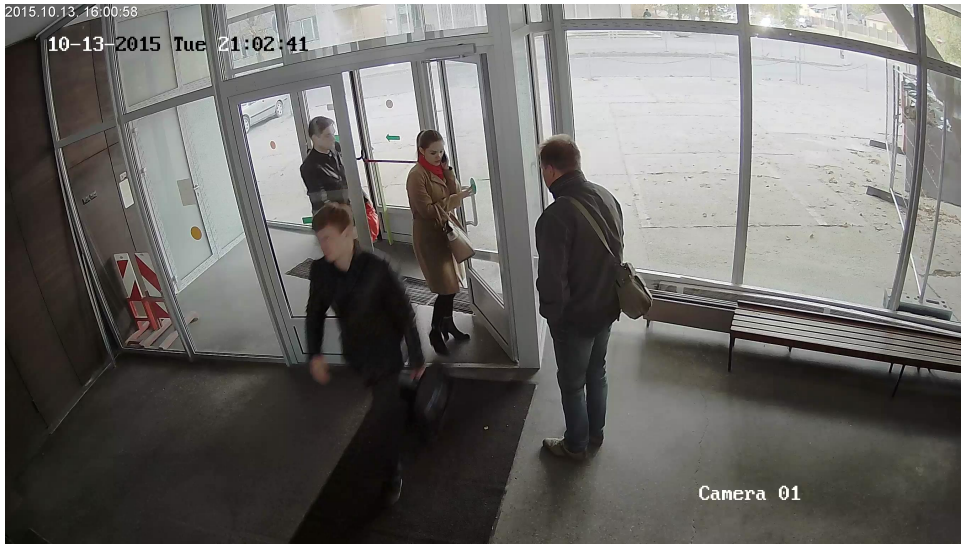


Rezultāts:
Klase atrasta uzreiz 52.27 %
Divas varbūtīgākās klases 68.51 %
3 varbūtīgākās klases 77.02 %
4 varbūtīgākās klases 83.16 %

Attēls → attēla apraksts



- **Neural-talk:** https://github.com/karpathy/neuraltalk2
- **Neural-storyteller:** https://github.com/ryankiros/neural-storyteller



1) A few boys ride their skateboards on the bridge.

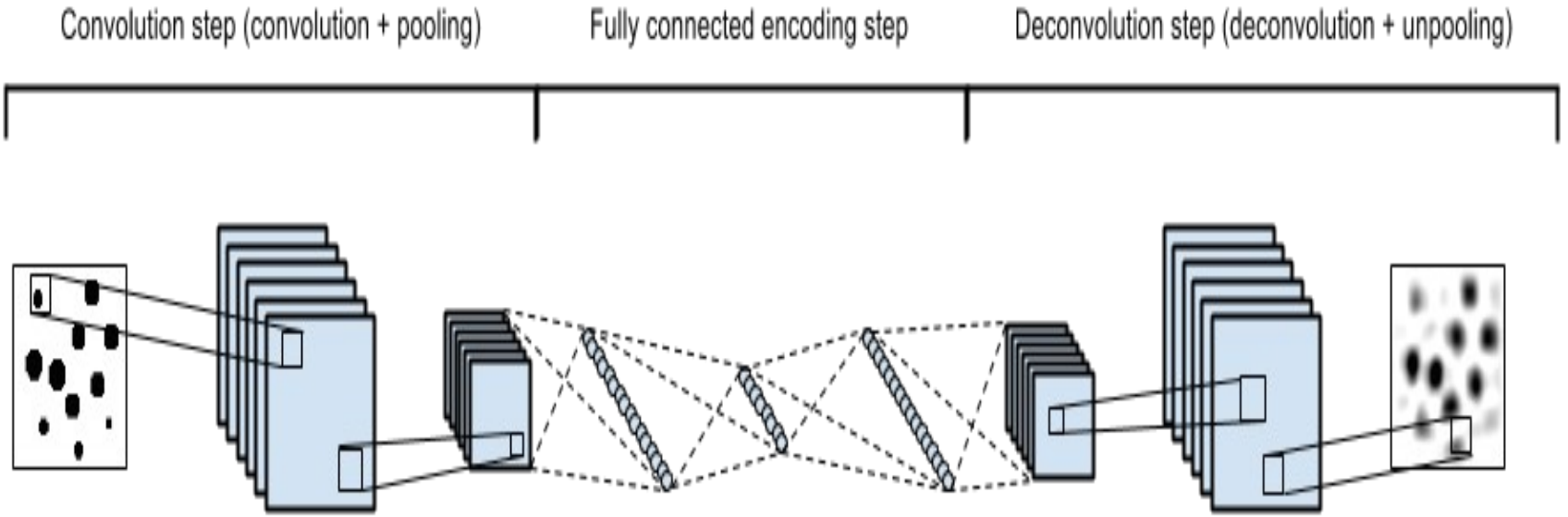
2) A group of people standing on the bridge.

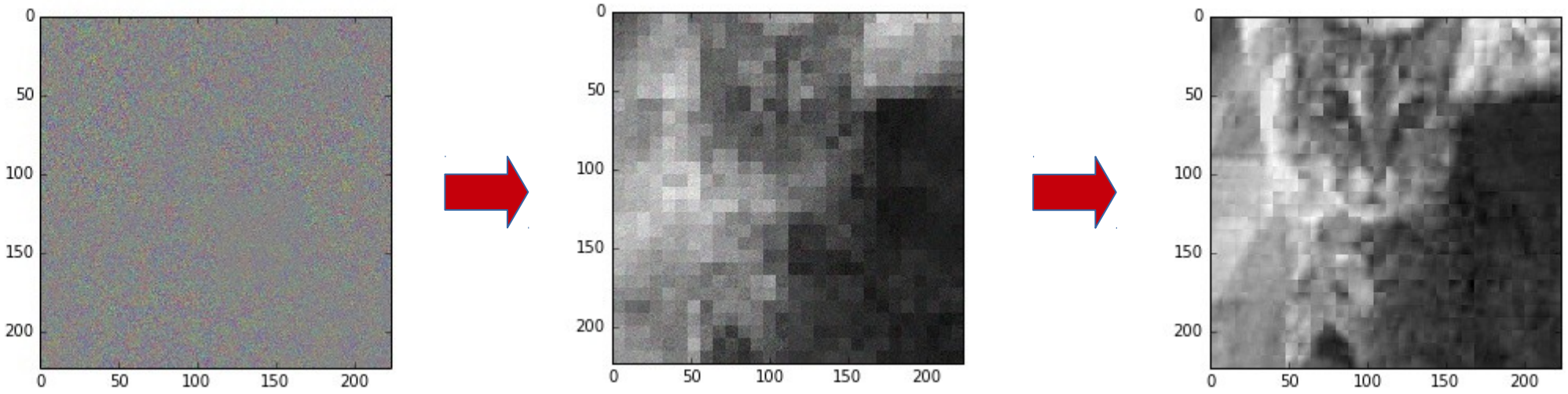
3) A couple of guys that are next to bridge

4) A group of young guys riding a skateboard across a bridge.

5) People walking on a bridge with much luggage

Sparse Auto-Encoder



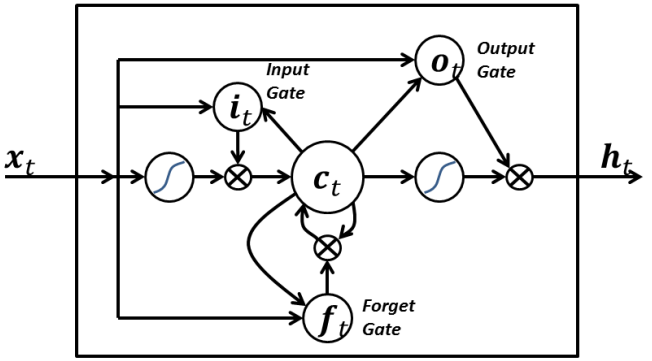


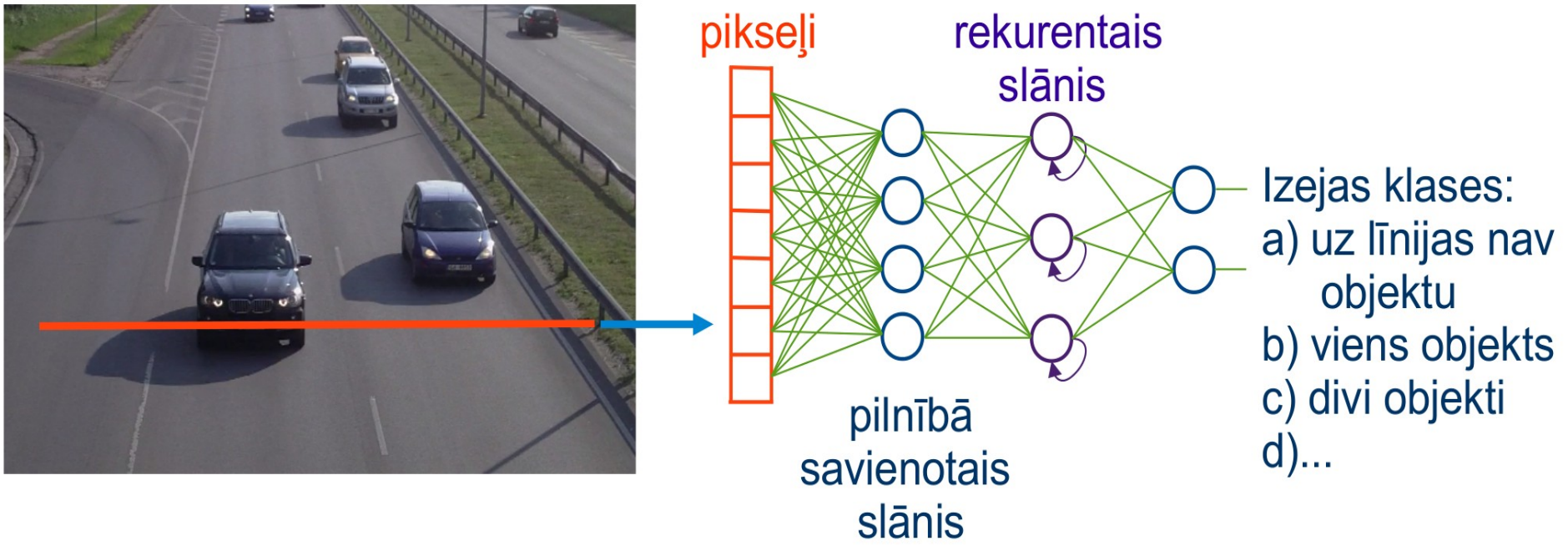
Kustīgu objektu atklāšana video

Algoritms

- Atklāšanas līnija kadrā – viena pikseļu rinda
- Neironu tīkls ar rekurentu slāni (RNN, LSTM)
- Tīkla izejā klasifikācija – cik objektu uz līnijas

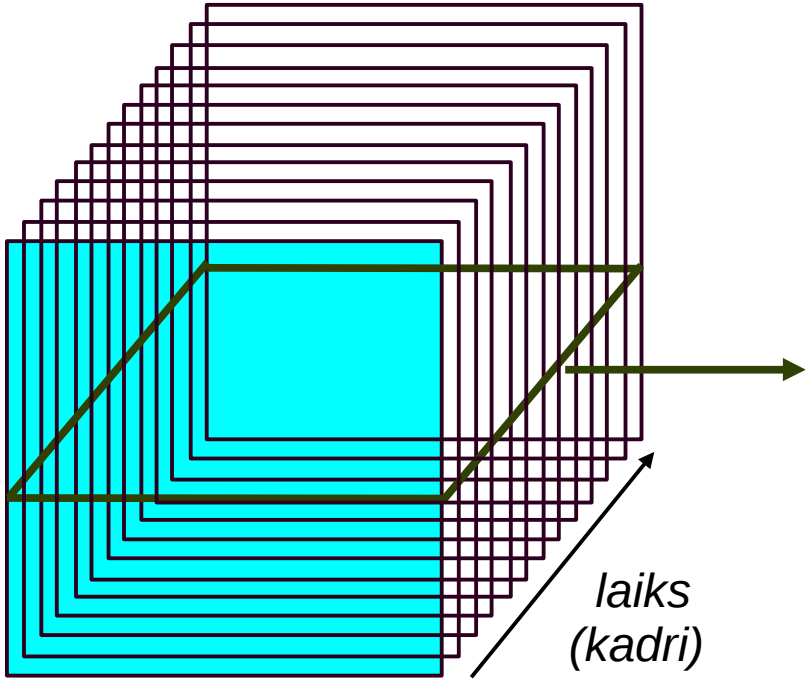
LSTM

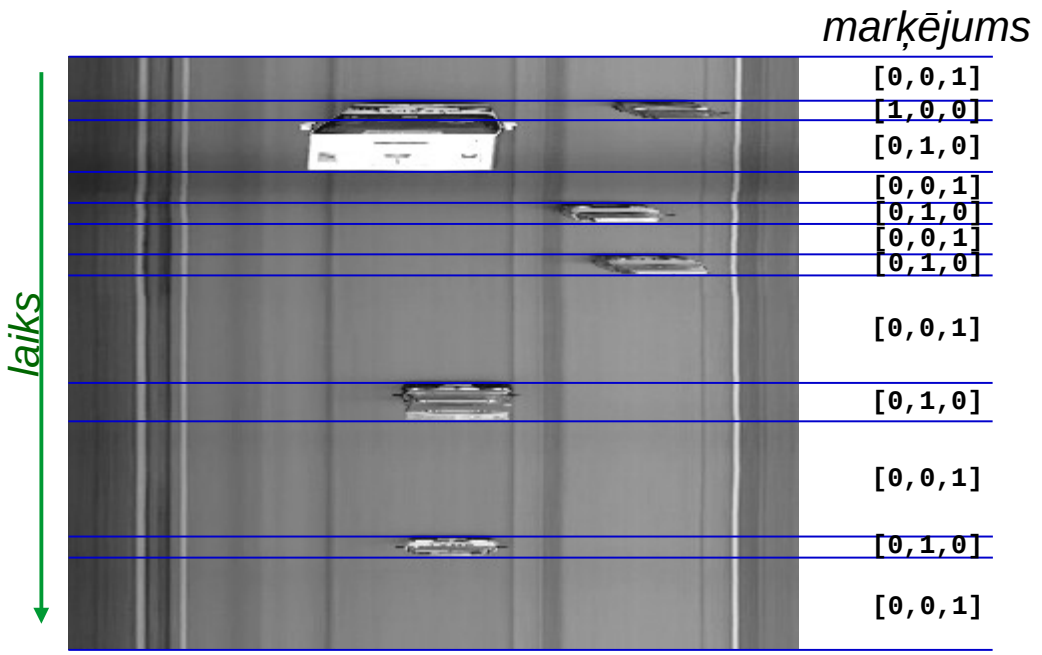




Telpas-laika attēls

- Atklāšanas līnijai katrā kadrā jāpiešķir atbilstošs marķējums:
  - [0, 0, 1] – klase 1 – uz līnijas nav kustīgu objektu
  - [0, 1, 0] – klase 2 – uz līnijas viens kustīgs objekts
  - [1, 0, 0] – klase 3 – uz līnijas divi kustīgi objekti

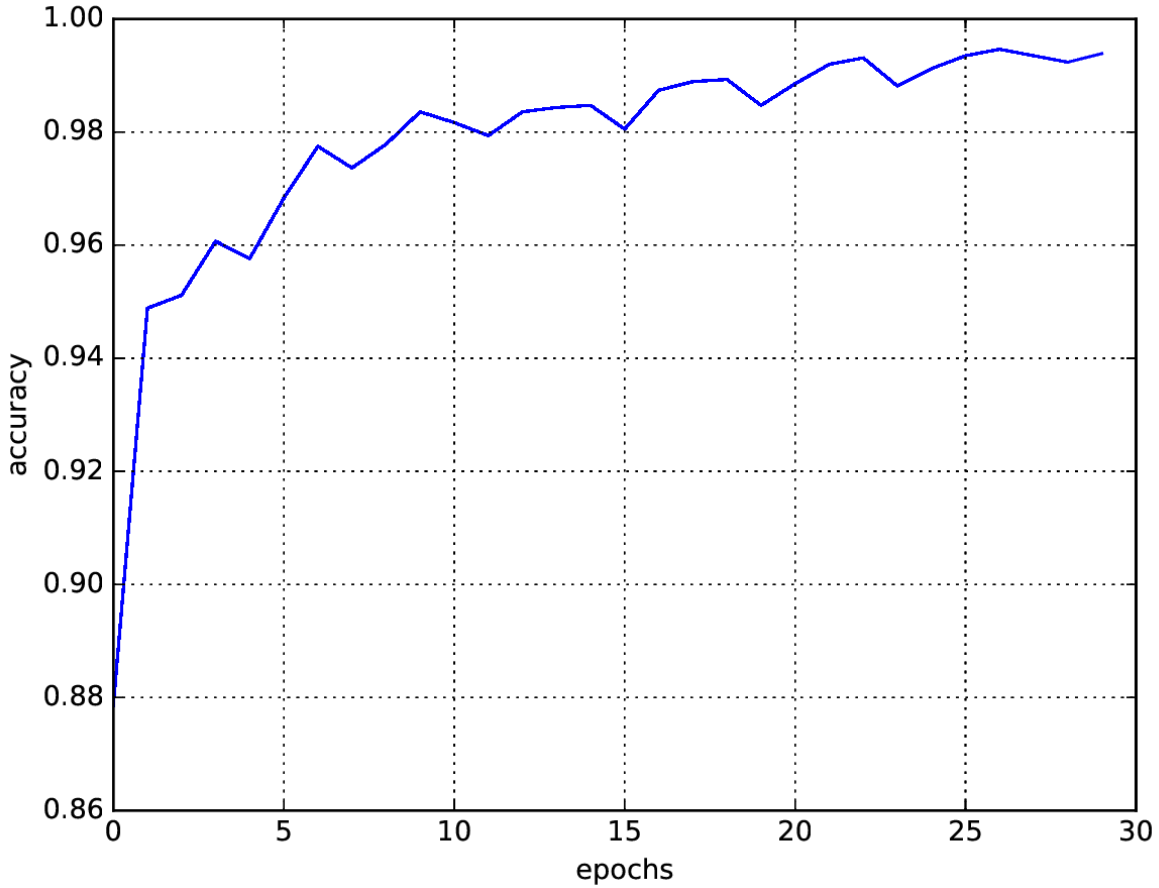




Marķēts attēls tīkla trenēšanai

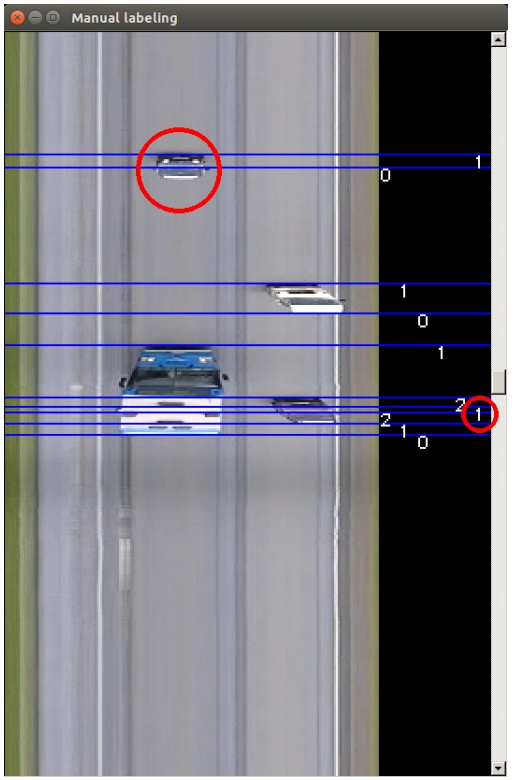
Algoritma apmācība

- Algoritms īstenots ar TensorFlow
- Tīkls trenēts ar vienu manuāli marķētu video



Datu marķēšanas programmas

- Pusautomātiska marķēšanas metode
  - Algoritms apstrādā ieejas video un veic automātisku marķēšanu
  - Grafiskā saskarne ļauj ērti noteikt un labot algoritma kļūdas



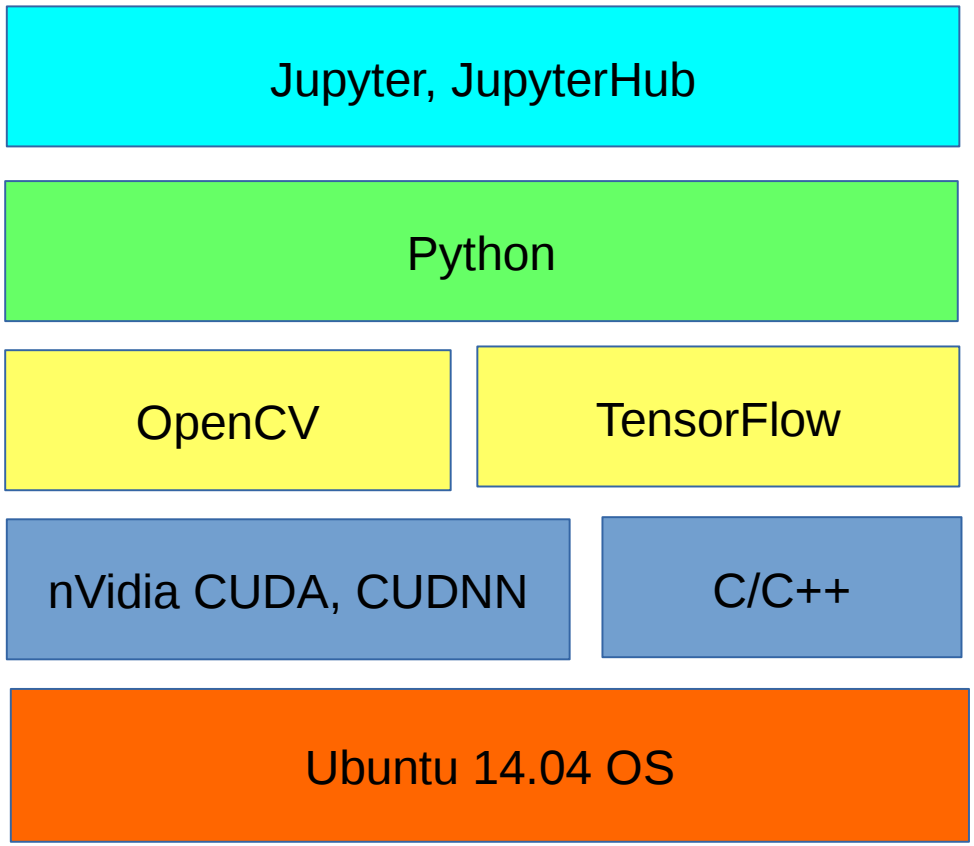
Skaitļošanas vide dziļās mašīnmācīšanās pielietojumiem

1 HPC serveris

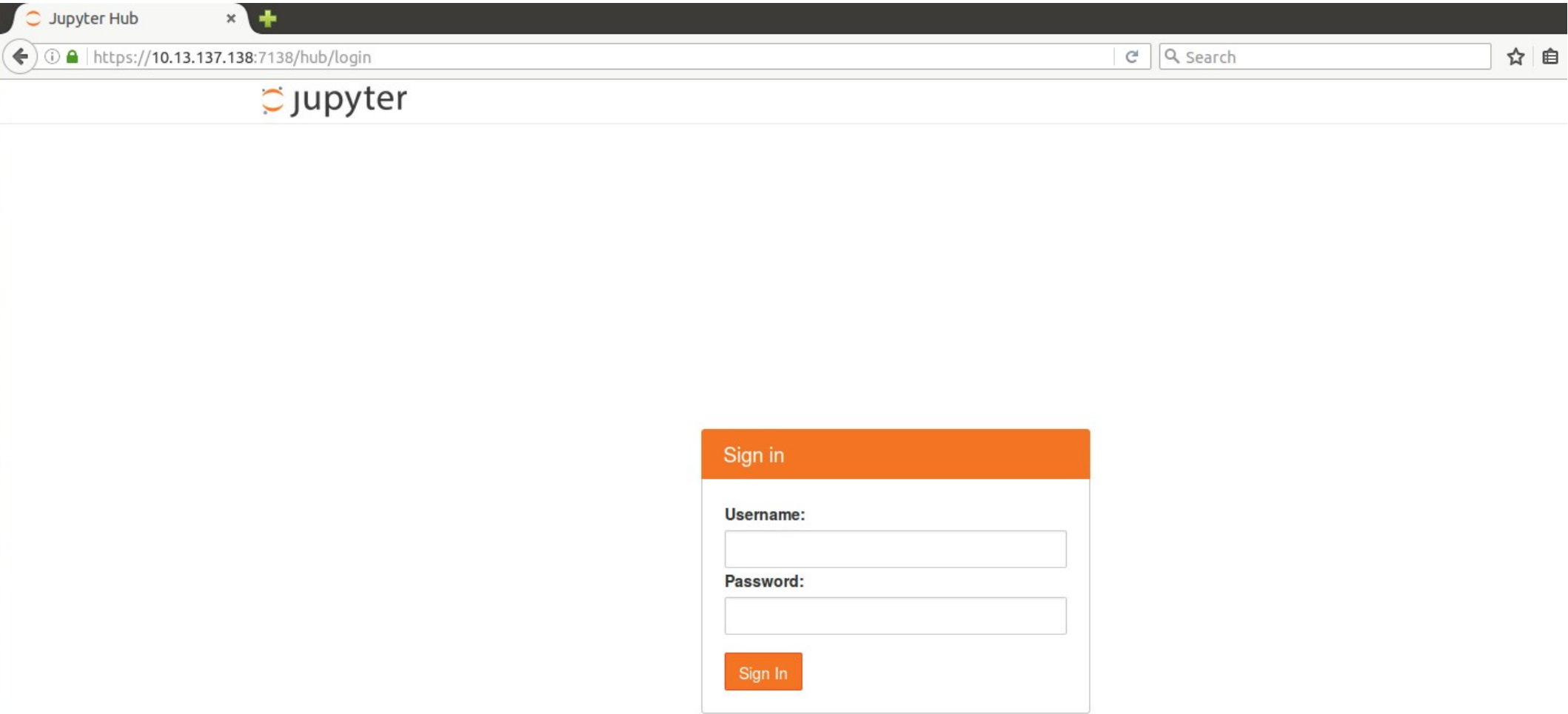
13 HPC serveru klāsteris

1 HPC serveris	HPC klāsteris
1 serveris	1 piekļuves un 12 skaitļošanas serveri
2x Intel Xeon CPU E5-2650 2.6 GHz	2x Intel Xeon CPU E5-2650 2.6 GHz
128 GB RAM	64 GB RAM
4x Nvidia Tesla K20 GPU (3.52 Tflops, 5 GB)	1x Nvidia Tesla K40 GPU (4.29 Tflops, 12 GB)

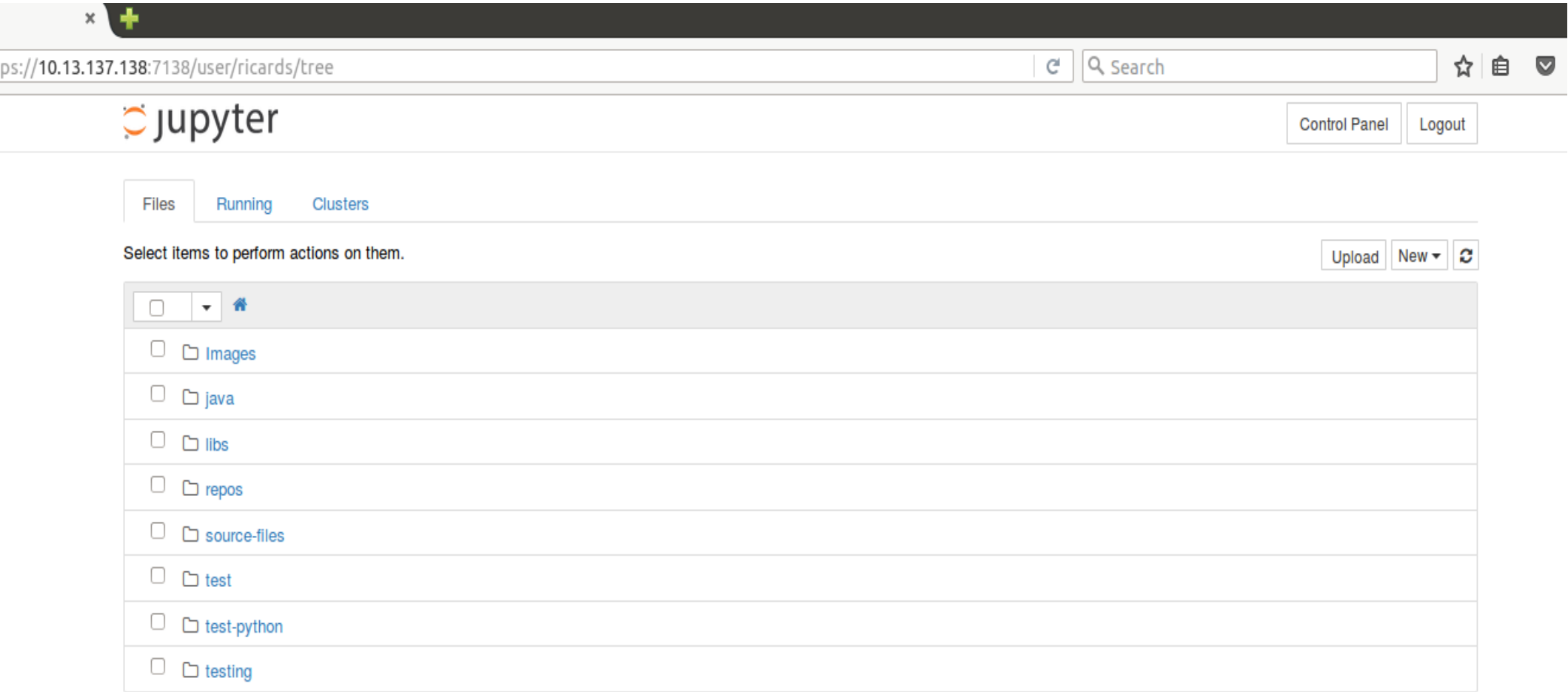
1 HPC Servera programmatūra



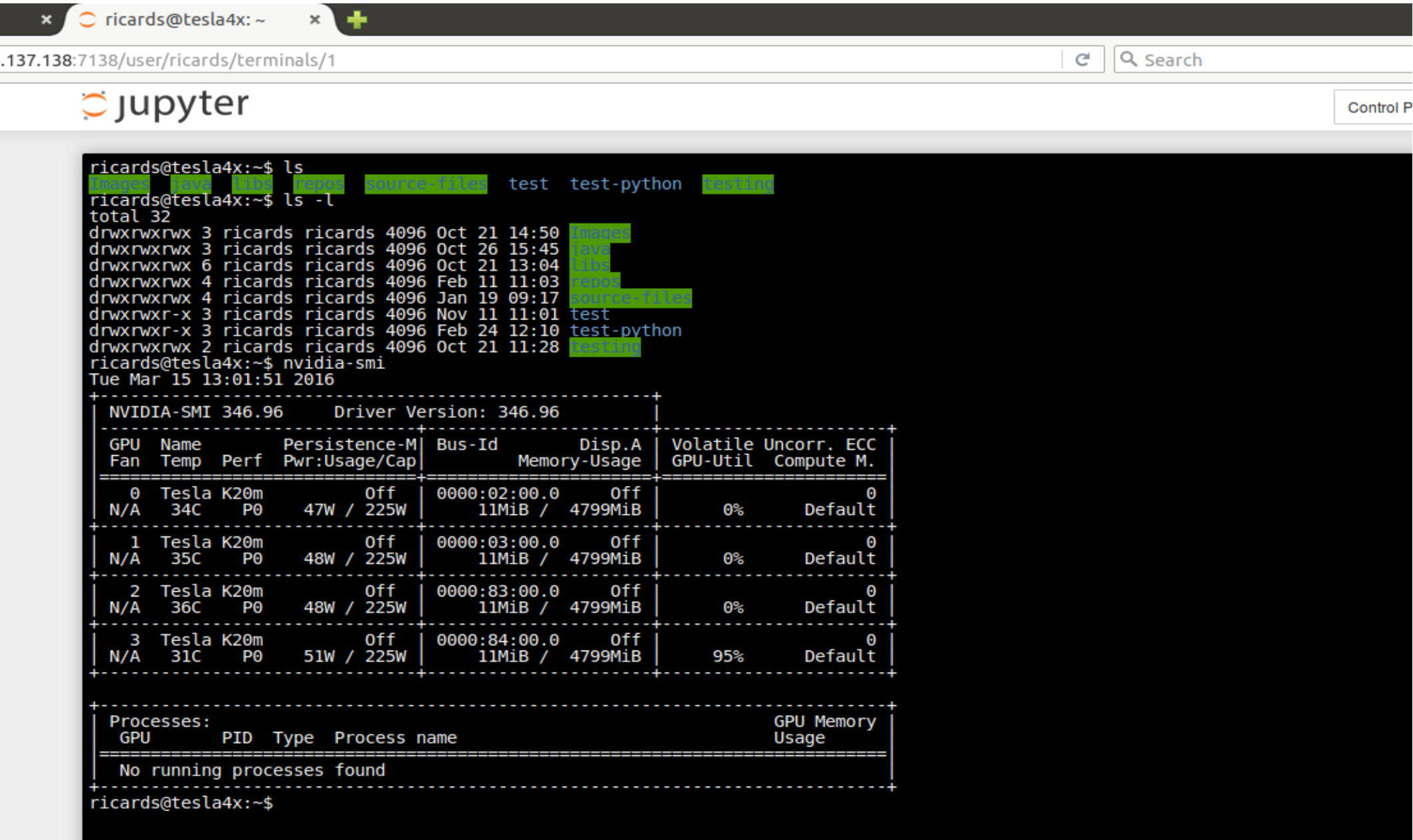
Piekļuve serverim (JupyterHub)



JupyterHub failu pārlūks



Jupyterhub linux terminālis



JupyterHub teksta redaktors

Untitled

10.13.137.138:7138/user/ricards/notebooks/Untitled.ipynb?kernel_name=python3

jupyter Untitled Last Checkpoint: 7 minutes ago (autosaved)

Control Panel Logout

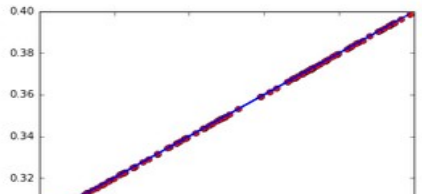
File Edit View Insert Cell Kernel Help Python 3

Code CellToolbar

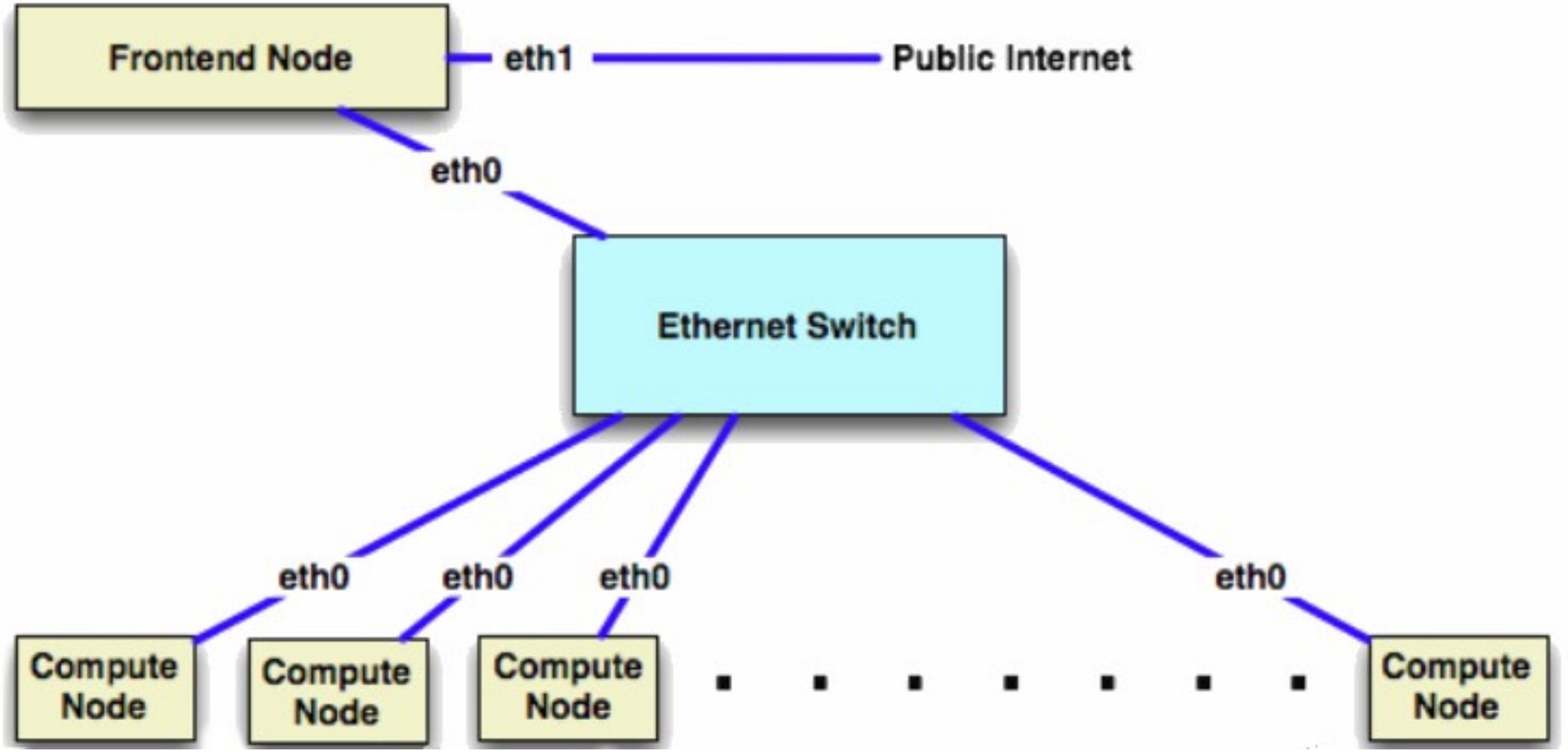
In [7]:
```
import tensorflow as tf
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

# Create 100 phony x, y data points in NumPy, y = x * 0.1 + 0.3
x_data = np.random.rand(100).astype(np.float32)
y_data = x_data * 0.1 + 0.3

# Try to find values for W and b that compute y_data = W * x_data + b
# (We know that W should be 0.1 and b 0.3, but Tensorflow will
# figure that out for us.)
W = tf.Variable(tf.random_uniform([1], -1.0, 1.0))
b = tf.Variable(tf.zeros([1]))
y = W * x_data + b

# Minimize the mean squared errors.
loss = tf.reduce_mean(tf.square(y - y_data))
optimizer = tf.train.GradientDescentOptimizer(0.5)
train = optimizer.minimize(loss)

# Before starting, initialize the variables.  We will 'run' this first.
init = tf.initialize_all_variables()

# Launch the graph.
sess = tf.Session()
sess.run(init)

# Fit the line.
for step in range(201):
    sess.run(train)
    if step % 20 == 0:
        Wi=sess.run(W)
        bi=sess.run(b)
        print(step, Wi, bi, sess.run(loss))
sess.close()
plt.plot(x_data, y_data, 'ro')
plt.plot(x_data, Wi*x_data+bi)
# Learns best fit is W: [0.1], b: [0.3]
```

```
0 [-0.15273017] [ 0.63787282] 0.0465723
20 [ -9.33366828e-05] [ 0.35624638] 0.000775203
40 [ 0.07021578] [ 0.31673694] 6.864e-05
60 [ 0.09113728] [ 0.30498031] 6.07769e-06
80 [ 0.09736278] [ 0.30148196] 5.38144e-07
100 [ 0.09921527] [ 0.300441] 4.76499e-08
120 [ 0.09976649] [ 0.30013123] 4.21942e-09
140 [ 0.09993052] [ 0.30003905] 3.73621e-10
160 [ 0.09997933] [ 0.30001163] 3.31053e-11
180 [ 0.09999385] [ 0.30000347] 2.9302e-12
200 [ 0.09999816] [ 0.30000106] 2.6489e-13
```

Out[7]: [<matplotlib.lines.Line2D at 0x7f97f2843f98>]



HPC klāstera arhitektūra



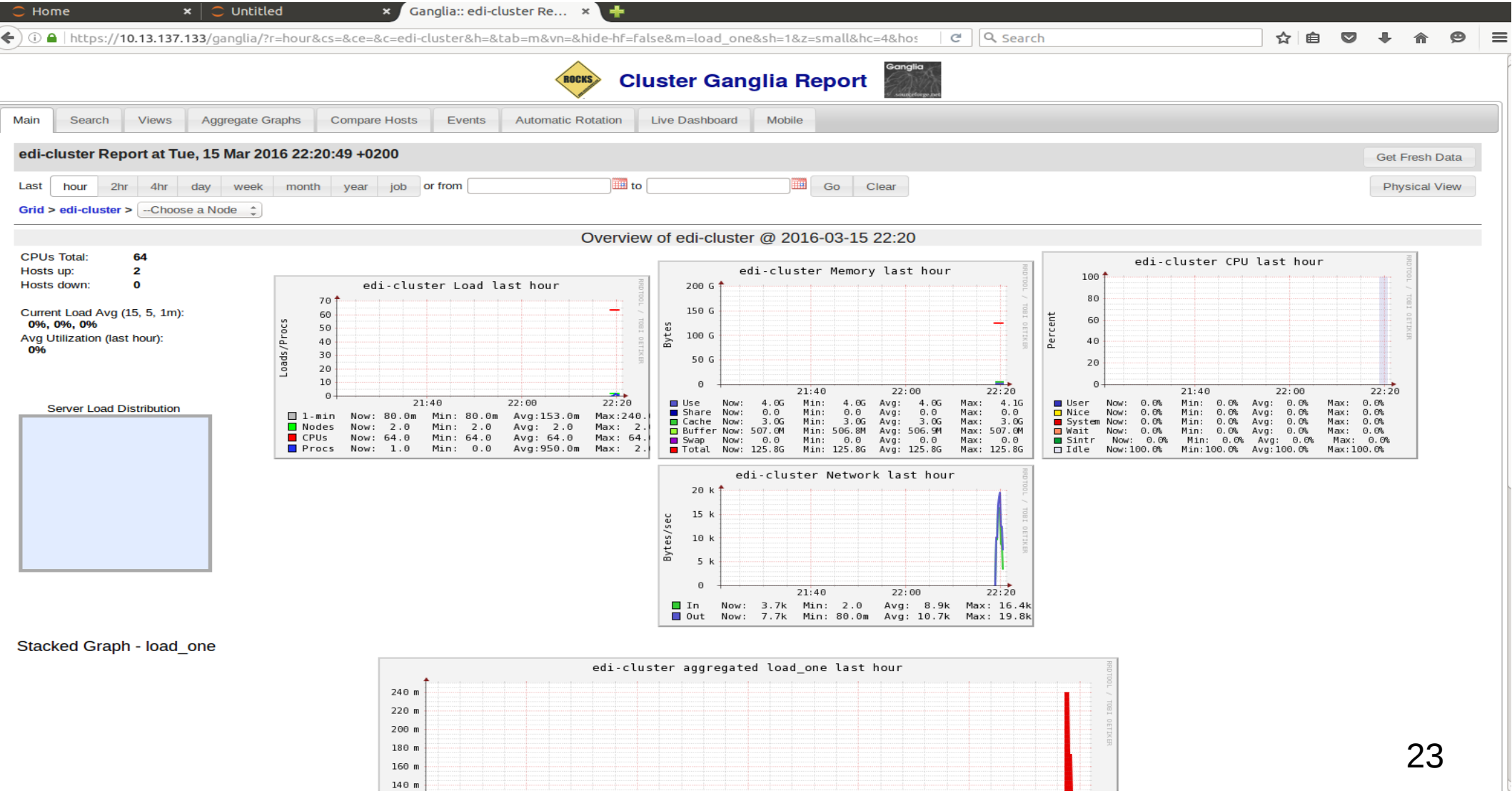
HPC klāstera programmatūra

- Linux Rocks 6.2 distribūcija (Balstīta uz CentOS)
  - Operētājsistēma, kas paredzēta augstas veiktspējas serveru klāsteru vidēm
  - Izstrādāta akadēmiskām vidēm ar mērķi padarīt klāsteru uzstādīšanu un uzturēšanu pēc iespējas vienkāršāku
  - Nodrošina ērtu mērogošanu (iespēju paplašināt klāsteri)

Linux Rocks programmatūra

- Distribūcija ietver sevī dažādas programmatūru pakas:
  - Ganglia – resursu monitoringa rīks
  - SGE (Oracle Grid Engine) – nodrošina izkliedētu procesu izpildi un pārraudzīšanu
  - Programmēšanas valodas: Java, Python, C/C++
  - Nepieciešams instalēt nVidia draiverus un CUDA
  - Nepieciešams uzstādīt dziļās mašīnmācīšanās bibliotēkas (Tensor Flow)

Ganglia resursu monitoringa rīks



Paldies!