Vilnius universitetas
Matematikos ir informatikos fakultetas
Informatikos katedra

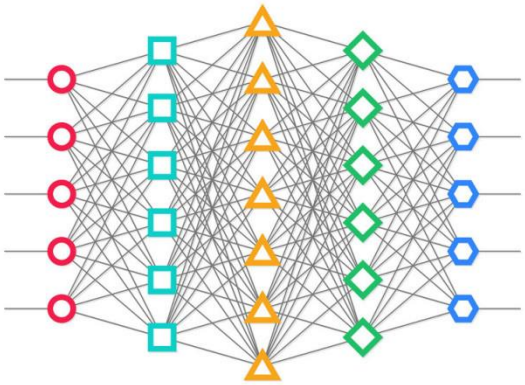
Dirbtiniai neuroniniai tinklai (programinė įranga)

prof. dr. Olga Kurasova
Olga.Kurasova@mii.vu.lt

2018

Neuroninių tinklų programinė įranga

- Pradėjus kurti dirbtinius neuroninius tinklus, atsirado poreikis kurti ir **programinę įrangą**, kurioje jei būtų įgyvendinti.
- Atsiradus naujiems neuroninių tinklų tipams ir architektūroms, programinė įranga turi būti nuolat **atnaujinama**.



Neuroniniai tinklai įvairiose sistemose

Neuroniniai tinklai įgyvendinti įvairiose duomenų tyrybos sistemose:

- **WEKA** – daugiasluoksnis perceptronas

- **Orange** – SOM tinklai

- **R** paketas

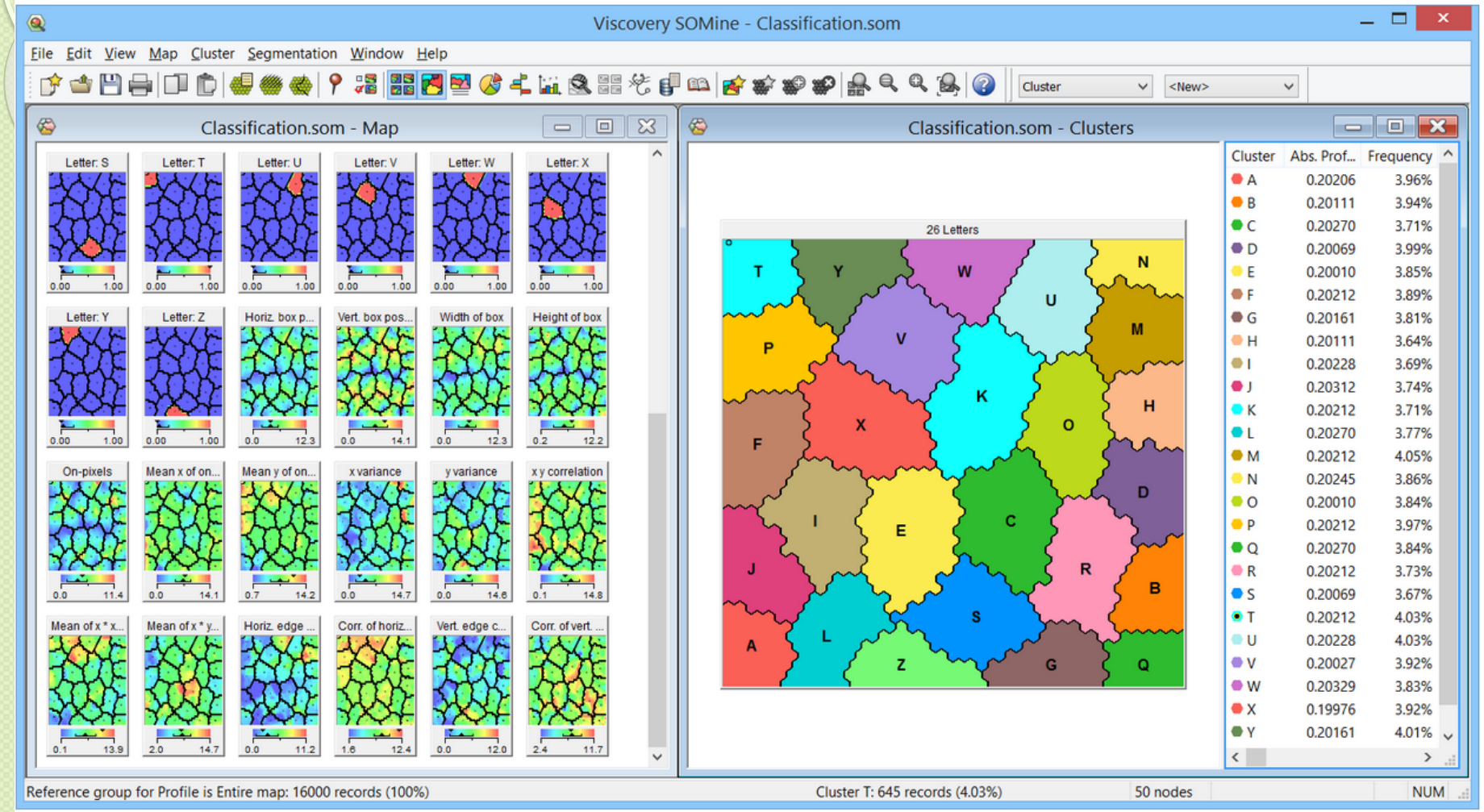
  - *library*(*neuralnet*) – daugiasluoksnis perceptronas
  - *library*(*nnet*) – daugiasluoksnis perceptronas
- **Matlab** – Neural Network Toolbox

Viscovery SOMine

- **Viscovery SOMine** is a workflow-oriented software suite based on **self-organizing maps** (SOM) and multivariate statistics for explorative data mining and predictive modeling.
- Main functions and features:
  - **Creation** of self-organizing map models using predefined schedules
  - Interactive **SOM visualization** and exploration
  - Visual cluster analysis with integrated visualization of **cluster boundaries** and inner structures
  - **Statistical functions**, such as descriptive statistics, histograms, correlations, PCA, and scatter plots

From: https://www.viscovery.net/somine/

Viscovery SOMine



Optical character recognition

From: https://www.viscovery.net/somine/

Gilieji neuroniniai tinklai R pakete

Table 1. List of available deep learning methods across the R packages.

PACKAGE	AVAILABLE ARCHITECTURES OF NEURAL NETWORKS
MXNetR	Feed-forward neural network, convolutional neural network (CNN)
darch	Restricted Boltzmann machine, deep belief network
deepnet	Feed-forward neural network, restricted Boltzmann machine, deep belief network, stacked autoencoders
H2O	Feed-forward neural network, deep autoencoders
deepr	Simplify some functions from H2O and deepnet packages

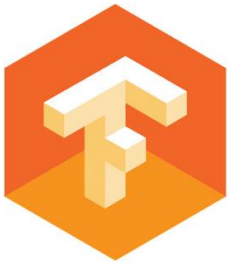
Iš: https://www.r-bloggers.com/deep-learning-in-r-2/

About TensorFlow

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.

Iš: https://www.tensorflow.org/

TenzorFlow



- Version 1.0.0 was released on **February 11, 2017**.
- **TensorFlow** is available on 64-bit Linux, macOS, Windows, and mobile computing platforms including Android and iOS.
- In May 2016, Google announced its **Tensor processing unit** (TPU), an ASIC built specifically for machine learning and tailored for TensorFlow.
- In May 2017, Google announced the **second-generation**, as well as the availability of the TPUs in Google Compute Engine. The second-generation TPUs deliver up to 180 teraflops of performance, and when organized into clusters of 64 TPUs, provide up to 11.5 petaflops.
- In February 2018, Google announced that they were making TPUs available in beta on the **Google Cloud Platform**.

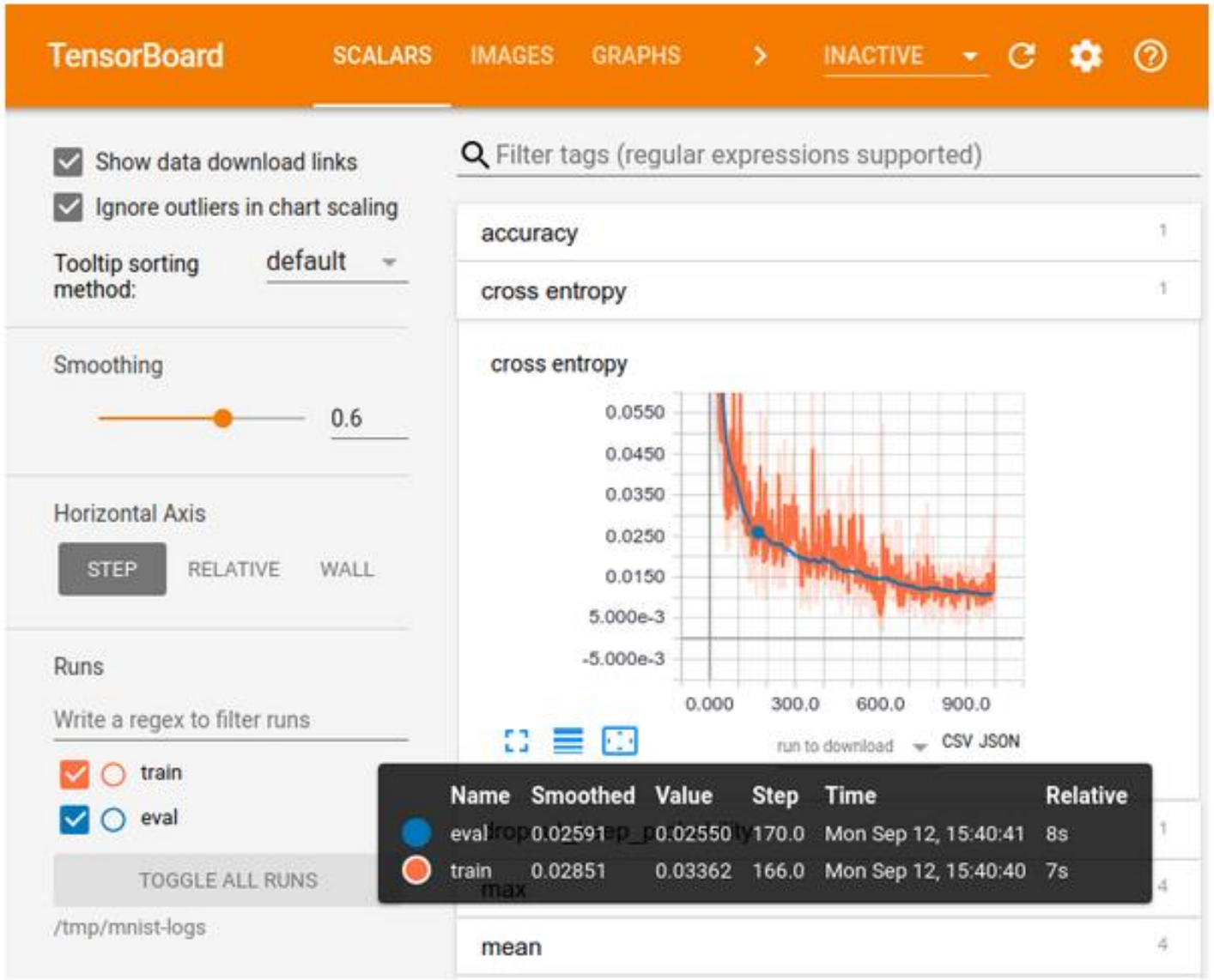
From: https://en.wikipedia.org/wiki/TensorFlow

TensorBoard

- **TensorBoard** – tai mokymo proceso vizualizavimo įrankis.
- Mokymo metu programiškai **fiksuojamos tikslumo reikšmės**, kurios atvaizduojamos TensorBoard įrankyje.

The computations you'll use TensorFlow for - like training a massive deep neural network - can be complex and confusing. To make it easier to understand, debug, and optimize TensorFlow programs, we've included a suite of visualization tools called TensorBoard. You can use TensorBoard to visualize your TensorFlow graph, plot quantitative metrics about the execution of your graph, and show additional data like images that pass through it. When TensorBoard is fully configured, it looks like this:

TensorBoard



Keras

- **Keras** is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano.
- If you want to quickly build and test a neural network with minimal lines of code, choose **Keras**. With Keras, you can build **simple or very complex neural networks** within a few minutes.

Deep Learning frameworks

Deep Learning frameworks operate at 2 levels of abstraction:

- Lower Level: This is where frameworks like Tensorflow, MXNet, Theano, and PyTorch sit. This is the level where mathematical operations like Generalized Matrix-Matrix multiplication and Neural Network primitives like Convolutional operations are implemented.
- Higher Level: This is where frameworks like Keras sit. At this Level, the lower level primitives are used to implement Neural Network abstraction like Layers and models. Generally, at this level other helpful APIs like model saving and model training are also implemented.

https://www.quora.com/Is-Keras-better-than-Tensorflow-for-deep-learning

Neuroniniai tinklai, TenzorFlow ir Keras

- **Konvoliuciniai** neuroniniai tinklai
- **Rekurentiniai** neuroniniai tinklai (LSTM – Long short-term memory)

Theano

- **Theano** is a Python library that allows you to define, optimize, and evaluate mathematical expressions involving multi-dimensional arrays efficiently.

Theano 1.0.0 (15th of November, 2017)

This is a final release of Theano, version `1.0.0`, with a lot of new features, interface changes, improvements and bug fixes.

Microsoft Cognitive Toolkit (CNTK)

The **Microsoft Cognitive Toolkit** (open source)—previously known as **CNTK**—empowers you to harness the **intelligence** within massive datasets through **deep learning** by providing uncompromised scaling, speed, and accuracy with commercial-grade quality and compatibility with the programming languages and algorithms you already use.

H2O

- **H2O** is an open source, in-memory, distributed, fast, and scalable machine learning and predictive analytics platform that allows you to build **machine learning** models on big data and provides easy productionalization of those models in an enterprise environment.
- **H2O**'s core code is written in **Java**.
- **H2O's REST API** allows access to all the capabilities of H2O from an external program or script via **JSON** over **HTTP**.

From: http://docs.h2o.ai/h2o/latest-stable/h2o-docs/welcome.html

H2O and Neural Networks

- H2O's Deep Learning is based on a multi-layer **feedforward artificial neural network** that is trained with stochastic gradient descent using back-propagation.
- The network can contain a **large number of hidden layers** consisting of neurons with tanh, rectifier, and maxout activation functions.
- **Advanced features** such as adaptive learning rate, rate annealing, momentum training, dropout, L1 or L2 regularization, checkpointing, and grid search enable high predictive accuracy.
- Each compute node trains a copy of the global model parameters on **its local data** with multi-threading (asynchronously) and contributes periodically to the global model via model averaging across the network.

H2O and Neural Networks

- The **H2O Deep Water project** supports **CNNs** and **RNNs** though third-party integrations of other deep learning libraries such as TensorFlow, Caffe and MXNet.

Caffe

- **Expression**: models and optimizations are defined as plaintext schemas instead of code.
- **Speed**: for research and industry alike speed is crucial for state-of-the-art models and massive data.
- **Modularity**: new tasks and settings require flexibility and extension.
- **Openness**: scientific and applied progress call for common code, reference models, and reproducibility.
- **Community**: academic research, startup prototypes, and industrial applications all share strength by joint discussion and development in a BSD-2 project.
- Web image classification demo http://demo.caffe.berkeleyvision.org/

Apache MXNet

- **MXNet** – a flexible and efficient library for deep learning.
- Based on the the Gluon API specification, the new **Gluon** library in **Apache MXNet** provides a clear, concise, and simple API for deep learning. It makes it easy to prototype, build, and train deep learning models without sacrificing training speed.

Torch

Torch is a scientific computing framework with wide support for machine learning algorithms that puts GPUs first. It is easy to use and efficient, thanks to an easy and fast scripting language, LuaJIT, and an underlying C/CUDA implementation.

A summary of core features:

- a powerful N-dimensional array
- lots of routines for indexing, slicing, transposing, ...
- amazing interface to C, via LuaJIT
- linear algebra routines
- neural network, and energy-based models
- numeric optimization routines
- Fast and efficient GPU support
- Embeddable, with ports to iOS and Android backends

From: http://torch.ch/

ANN software (more)

neuraldesigner

Neuroph
Java Neural Network Framework

NeuroSolutions

simbrain

GMDH

Ką rinktis?

- Ar norima **paprastos** ir **vartotojui draugiškos**?
  - *desktop application*
- Ar bus vykdomi **intensyvūs skaičiavimai**?
  - GPU ar debesija pagrįsti sprendimai
- Ar norima patiems **programuoti**?
  - Pageidautina programavimo kalba
- Ar norima vykti **išsamius tyrimus**?
  - Patogus būdas modifikuoti, automatizuoti.

Jupyter notebook

- The **Jupyter Notebook** is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text.
- Uses **include**: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, **machine learning**, and much more.

http://jupyter.org/index.html

Jupyter notebook

Galima vykdyti kodą iš karto **web aplinkoje**.

In [2]:
```
# Load the data
train = pd.read_csv("../input/train.csv")
test = pd.read_csv("../input/test.csv")
```

In [3]:
```
Y_train = train["label"]

# Drop 'label' column
X_train = train.drop(labels = ["label"],axis = 1)
```