

# Rozpoznávanie obrazcov - 10. cvičenie

## Neurónové siete

Viktor Kocur  
viktor.kocur@fmph.uniba.sk

DAI FMFI UK

16.4.2019

# Keras

## Keras

V dnešnom cviku budeme používať keras, ktorý poskytuje pekné API na trénovanie neurónových sietí. Je možné ho použiť s rôznymi backendmi, ale my využijeme tensorflow. Dnes ho použijeme, ale na cloude. Ak si ho chcete nainštalovať na svoj počítač, tak stačí použiť `pip install keras`.

## Tensorflow

Inštalácia tensorflowu môže byť komplikovanejšia (hlavne ak chcete GPU support). Je možné sa pozrieť na link <https://www.tensorflow.org/install>.

# Google Colab

## Google Colab

Google Colab je cloudová služba, ktorá je zadarmo a umožňuje spúšťať ipython notebooky aj na GPU a TPU. Nájsť ho môžete na stránke <https://colab.research.google.com/>

## Notebooky

Notebook si môžete vytvoriť vlastný, alebo sa pozrieť na príklady, ktoré už sú v google colab. Pre toto cvičenie si môžete otvoriť notebooky z môjho githubu. Môj username je kocurvik.

# Google Colab

Welcome to Colaboratory!

free Jupyter notebooks  
you can run in the cloud

ing Colab  
ute video

et else

nt  
Go

Cod

rted

u are reading is a [Jupyter notebook](#), hosted in Colaboratory. It is not a static page, but an interactive environment that lets you write and execute code in other languages.

EXAMPLES

RECENT

GOOGLE DRIVE

GITHUB

UPLOAD




Enter a GitHub URL or search by organization or user




☐ Include private repos


kocurvik

Repository: [🔗](#) kocurvik/edu Branch: [🔗](#) master

Path

 RO/materialy/cv10/mnist\_run.ipynb  

 RO/materialy/cv10/mnist\_train.ipynb  

NEW PYTHON 3 NOTEBOOK  CANCEL

# Google Colab

## Unixové príkazy

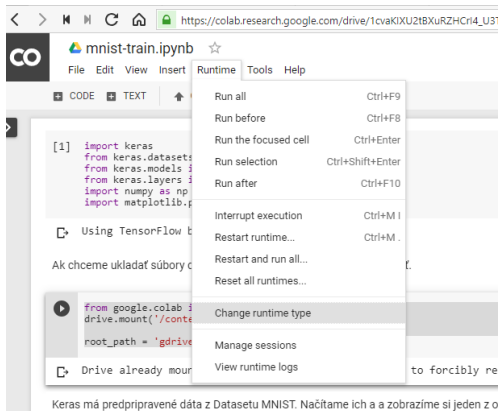
Pracovať môžeme ako s normálnym ipython notebookom. Pred príkazy ktoré chceme spustiť mimo python (cd, ls, mkdir, atď.) stačí pridať výkričník.

## Google Drive

Google colab notebook sa dá ľahko pripojiť ku Google drive:

```
from google.colab import drive
drive.mount('/content/gdrive')
root_path = 'gdrive/My Drive/Colab/'
```

# GPU support



# Plne prepojená sieť

## MNIST

Plne prepojenú sieť si otestujeme na datasete MNIST, ktorý obsahuje obrázky ručne písaných číslíc v PSČ kolónkach.

## Notebook

Z githubu [https://github.com/kocurvik/edu/blob/master/R0/materialy/cv10/mnist\\_train.ipynb](https://github.com/kocurvik/edu/blob/master/R0/materialy/cv10/mnist_train.ipynb) si v Colabe otvorte notebook. A spolu si ho prejdeme.

## Úloha

Otestujte si rôzne optimalizačné algoritmy. To dosiahnete pomocou zmeny vo volaní `model.compile()`. Stačí prepísať string, ale ak chcete meniť aj parametre je nutné vyrobiť si objekt ktorý predstavuje optimalizátor. Pozrite si dokumentáciu kerasu a skúste použiť SGD s inými parametrami.

# Konvolučná neurónová sieť

## CIFAR-10

Tento krát použijeme dataset malých farebných obrázkov.

## Notebook

Z githubu [https://github.com/kocurvik/edu/blob/master/R0/materialy/cv10/cifar\\_train.ipynb](https://github.com/kocurvik/edu/blob/master/R0/materialy/cv10/cifar_train.ipynb) si v Colabe otvorte notebook. A spolu si ho prejdeme.

## Úloha

Skúste pridať vrstvy ako dropout a batchnorm. Pohrajte sa s architektúrou a optimizátormi. Využite dokumentáciu kerasu.

## Úloha

Vytvorte si notebook cifar\_run, ktorý spustí natrénovanú sieť na testovacie dáta.



# Transfer learning

## CIFAR-10

Teraz použije trocha absurdný dataset walk/run. Ale tak bude nám stačiť. Cieľ je zistiť či na obrázku niekto kráča, alebo beží.

## Notebook

Z githubu [https://github.com/kocurvik/edu/blob/master/R0/materialy/cv10/walkrun\\_transfer.ipynb](https://github.com/kocurvik/edu/blob/master/R0/materialy/cv10/walkrun_transfer.ipynb) si v Colabe otvorte notebook. A spolu si ho prejdeme.

## Úloha

Akoby ste postupovali ak by ste chceli trénovať celú sieť? Ako by ste vedeli na extrahované features použiť napr. SVM?

# NumPy

## NumPy

Pythonovská knižnica na manipuláciu s maticami a viac-rozmernými poliami. Dokáže viacere potrebné manipulácie. Mnoho ďalších knižníc využíva NumPy.

## import

Odteraz predpokladajme, že sme importovali numpy ako np:

```
import numpy as np
```

# np.ndarray

## np.ndarray

np.ndarray je základná trieda numpy. Tento objekt predstavuje vždy viac-rozmerné pole.

### Konštruktor

```
a = np.array([[1, 2, 3], [10, 20, 30]])
```

```
b = np.array([4, 5], dtype=np.uint8)
```

```
a.shape
```

```
a.dtype
```

```
b.dtype
```

### Zmena typu

```
d = a.astype(np.float64)
```

# Indexácia

Indexácia je veľmi podobná matlabu, rozdiely sú v podstate rovnaké ako medzi čistým pythonom a matlabom.

```
r = np.random.random((6,10,3))  
r[3,4,1]  
r[:, :, 1]  
r[0:4, 5:6, :]  
r[1::2, :, :]
```

Niekedy budeme potrebovať pridať singleton dimenziu. To sa robí pomocou `None`, alebo `np.newaxis` namiesto indexu.

```
r[0, :, :].shape  
r[None, 0, :, :].shape  
r[np.newaxis, 0, :, :].shape
```

# Broadcasting

## Tvorba arrays

```
o = np.ones((5,4))  
z = np.zeros(5, dtype=np.int8)
```

Broadcasting Podobne ako v matlabe aj v NumPy funguje broadcasting.

```
r += 10  
r[0,:,:] = np.random.random((10,3))  
r[0] = np.random.random((10,3))  
r[0] = np.zeros((10,1))  
r /= 500
```

# Manipulácia

## np.reshape

`np.reshape(arr, shape)` - vráti nové pole s tvarom podľa `shape`, `shape` je tuple, môže v ňom byť `None` pre dimenziu ktorej nevieme veľkosť dopredu

## np.concatenate

`np.concatenate((a1, a2, ...), axis=0)` - vráti spojenie polí `a1`, `a2` atď' pozdĺž dimenzie `axis`

## np.stack

`np.stack((a1, a2, ...), axis=0)` - vráti spojenie polí `a1`, `a2`, atď', tak že im vytvorí novú dimenziu