Lab: CudaVision – Learning Vision Systems on Graphics Cards (MA-INF 4308)

Assignment 1

1.6.2016

Prof. Sven Behnke
Dr. Seongyong Koo

universitätbonn ais

Why TensorFlow?

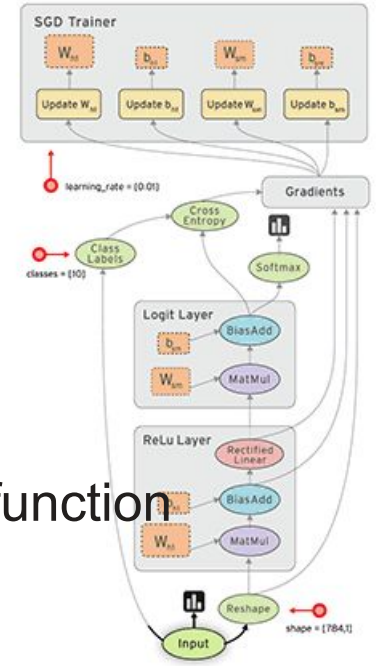
Deep learning software list

- **Theano** – CPU/GPU symbolic expression compiler in python
- **Torch** – provides a Matlab-like environment for state-of-the-art machine learning algorithms in lua
- **Tensorflow** – TensorFlow™ is an open source software library for numerical computation using data flow graphs. (python-based)
- **Caffe** – Caffe is a C++ deep learning framework made with expression, speed, and modularity in mind. (Berkeley Vision and Learning Center (BVLC) and by community contributors)
- **Keras** – A theano based high-level deep learning library, compatible to TensorFlow
- **CUV** – C++ framework with python bindings for easy use of Nvidia CUDA functions on matrices. It contains an RBM implementation, as well as annealed importance sampling code and code to calculate the partition function exactly (from AIS lab at University of Bonn).

universitätbonn ais

[http://deeplearning.net/software_links/]

Why TensorFlow?



Deep flexibility
- Express computation as a data flow graph
- Provide tools to assemble subgraphs
- Can write a higher-level libraries on top of TensorFlow
- Defining a new compositions of operators as a Python function
- Writing a C++ to add a new low-level data operator

- **True portability**
  - Running on CPUs or GPUs, and on desktop, server, or mobile platform
  - Ready to scale-up and faster on GPUs with no code changes

- **Connect Research and Production**
  - Industrial researchers to push ideas to producs faster
  - Academic researchers to share code more directly and with greater scientific reproducibility

universitätbonn ais

[https://www.tensorflow.org/]

Why TensorFlow?

- **Auto-differentiation**
  - With defining a predictive model with an objective function and data, TensorFlow handles computing derivatives automatically
  - Computing the derivative of some values w.r.t. other values in the model allows to extend your graph easily

- **Language options**
  - Easy to use Python interface and C++ interface
  - Interactive iPython notebook for research/education purposes

- **Maximize performance**
  - Freely assign compute elements of your Tensorflow graph to different devices (multiple CPUs and GPUs)

[https://www.tensorflow.org/]

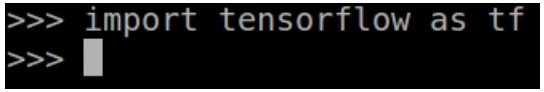
Easy Installation with Anaconda

- **For more details or other installation ways**
  - https://www.tensorflow.org/versions/r0.8/get_started/os_setup.html#download-and-setup

- **Installation with Anaconda**
  - Download Anaconda2 for Python 2.7 or Anaconda3 for Python 3.5
    - https://www.continuum.io/downloads
  - Run in terminal
    - `bash Anaconda2-4.0.0-Linux-x86_64.sh`
    - `conda install -c https://conda.anaconda.org/jjhelmus tensorflow`

- **Installed in cuda8 - 11, bigcuda 1,3,4 (directly or using ssh)**
  - Login with informatik ID
  - First try out TF in python
  - Or with Jupyter

    `$ jupyter notebook`

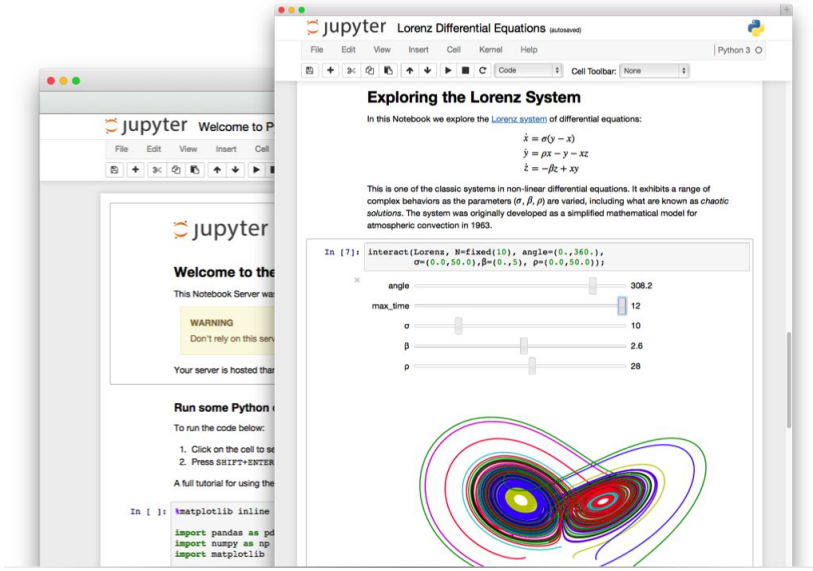


[https://www.tensorflow.org/]

Jupyter notebook

- **Interactive web application for python**
  - code, equations, visualizations, text
  - Sever and web client

- **Jupyter notebook for local user**
  - `$ jupyter notebook`
  - With a local user account, http://localhost:8888
  - See demo



[http://jupyter.org/]

Jupyter notebook

- **Jupyter notebook for remote user**
  - http://jupyter-notebook.readthedocs.io/en/latest/public_server.html
  - In short
    - `$ jupyter notebook --generate-config`
    - `$ python`

```
>>> from notebook.auth import passwd
>>> passwd()
Enter password:
Verify password:
'sha1:b9eee...'
>>> exit()
```

    - `$ vi /yourhome/.jupyter/jupyter_notebook_config.py`
    - Uncomment and modify these options

```
c.NotebookApp.ip = '*'
c.NotebookApp.password = u'sha1:b9eee...'
c.NotebookApp.open_browser = False
c.NotebookApp.port = 9999
```

any port number

  - You can access to the serve using http://pcname:portname

[http://jupyter.org/]

TensorFlow Basics

- **What is Tensor**
  - n-dimensional arrays (Vector: 1-D tensor, Matrix: 2-D tensor)
  - Deep learning (many machine learning) process are flows of tensors
  - A sequence of tensor operations

- **Define tensors as variables**
  - Variable: in-memory buffers containing tensors

```
w = tf.Variable(tf.random_normal([3, 3]), name='w')
```

    - Initialization
      - `tf.random_normal(), tf.zeros(), tf.ones() ...`

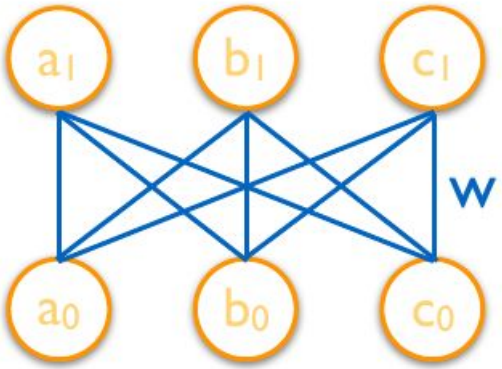
- **Tensor operations**
  - All functions in TensorFlow `var = tf.xxx(var, var, ...)`
  - math(`tf.add, tf.cos`, ...), image(`tf.image.x`), neural network (`tf.nn.x`),
  - `Variable` is one of operators

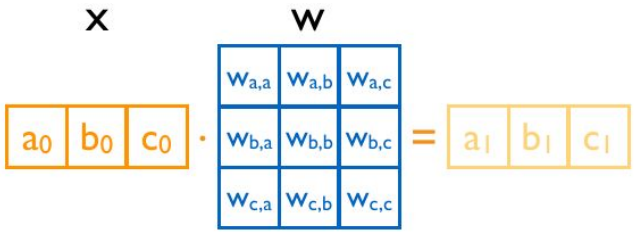
universitätbonn ais

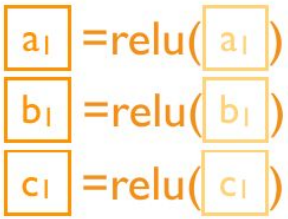
[https://www.tensorflow.org/versions/r0.8/how_tos/index.html]

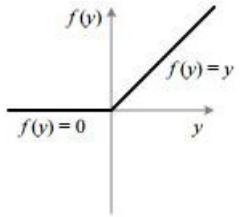
TensorFlow Basic Example

- **A simple Rectified Linear Unit (ReLU) network**










```
y = tf.matmul(x,w)
out = tf.nn.relu(y)
```

- Variable stores the state of current execution
  - `x`: input variable
  - `w = tf.Variable(tf.random_normal([3, 3]), name='w')`
  - `y, out` are automatically determined

[https://www.tensorflow.org/versions/r0.8/how_tos/index.html]

TensorFlow Basic Example

- **Session**
  - Manage resource for execution
  - A session should be defined before resource assignment

```
sess = tf.Session()
```

- **Input placeholder**
  - Assign a memory space for input data

```
x = tf.placeholder("float", [1, 3])
```

- **Code so far**

```
import tensorflow as tf
sess = tf.Session()
x = tf.placeholder("float", [1, 3])
w = tf.Variable(tf.random_normal([3, 3]), name='w')
y = tf.matmul(x, w)
relu_out = tf.nn.relu(y)
```

[https://www.tensorflow.org/versions/r0.8/how_tos/index.html]

TensorFlow Basic Example

- **Our ReLU network in TensorBoard**

```
import tensorflow as tf
sess = tf.Session()
x = tf.placeholder("float", [1, 3])
w = tf.Variable(tf.random_normal([3, 3]), name='w')
y = tf.matmul(x, w)
relu_out = tf.nn.relu(y)
writer =
  tf.train.SummaryWriter('/tmp/tf_logs/relu',sess.graph)

$ tensorboard --logdir=/tmp/tf_logs/relu
```

[https://www.tensorflow.org/versions/r0.8/how_tos/index.html]

TensorFlow Basic Example

- **Fetch**
  - Retrive content from a node
  - Variables should be filled beforehand

```
import numpy as np
import tensorflow as tf
sess = tf.Session()
x = tf.placeholder("float", [1, 3])
w = tf.Variable(tf.random_normal([3, 3]), name='w')
y = tf.matmul(x, w)
relu_out = tf.nn.relu(y)
sess.run(tf.initialize_all_variables())
print sess.run(relu_out, feed_dict={x:np.array([[1.0,
  2.0, 3.0]])})
```

- **Needs to release resource after use**

```
sess.close()
```

universitätbonn ais

[https://www.tensorflow.org/versions/r0.8/how_tos/index.html]

Assignment 1

- **Set Jupyter server in your home account so that you can use it remotely**

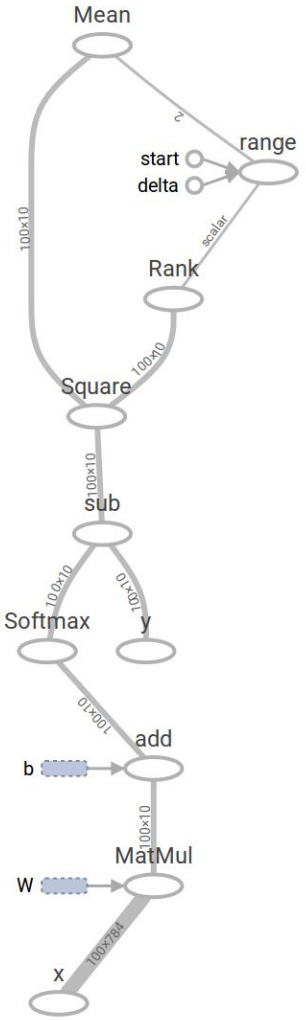
- **Build a network for Logistic Regression Classifier**
  - Input data x : 100 x 784
  - Input label y : 100 x 10
  - Weight Variable W with random_normal initialization
  - Bias variable b with zeros

$$activation = softmax(x \cdot W + b)$$

$$cost = MSE(activation - y)$$

  - Operators in use
    `tf.nn.softmax(), tf.matmul(), tf.reduce_mean(), tf.square()`

- **Submit your ipython notebook file**



[https://www.tensorflow.org/versions/r0.8/how_tos/index.html]