Introduction to Theano

Razvan Pascanu
Professeur: Yoshua Bengio

January 22, 2012

Université de Montréal

- The Tools
  - Git and Assembla
  - How to install Theano
  - The list of tools
- Theano
  - Theano at a Glance
  - Numpy refresher
  - Adding two numbers
  - Computing derivatives
  - Shared variables
  - Details of theano.function
  - Basic debugging
  - Basic profiling
- Deep Learning Tutorials
  - Putting it all together: Logistic Regression
- Q&A

Laboratoire d'Informatique

des Systèmes Adaptatifs

http://www.iro.umontreal.ca/~lisa

Université de Montréal

- Repo: `http://www.assembla.com/spaces/ift6266h12/wiki`

TODO:

- Make an assembla account
- Join the team (you can send me the username at `pascanur@iro.umontreal.ca`)
- Create a fork of the repository
- Learn to properly use Git

Université de Montréal

RTFM

`http://deeplearning.net/software/theano/install.html`

Université de Montréal

- Mandatory
  - Assembla [www.assembla.com]
  - Git [www.git-scm.com]
  - jobman, jobdispatch [www.deeplearning.net/software/jobman]
  - NumPy [www.numpy.org]
  - python [www.python.org]
  - psql [http://www.postgresql.org]
  - SciPy [www.scipy.org]
  - Theano [www.deeplearning.net/software/theano]
- Optional
  - ipdb [pypi.python.org/pypi/ipdb]
  - IPython [www.ipython.org]
  - matplotlib [http://matplotlib.sourceforge.net/]

Université de Montréal

```
1  # Compute sum_{i=0}^{N} x_i^2
2  import theano, theano.tensor as TT
3  x = TT.vector()
4  rval = (x ** 2).sum()
5  fn = theano.function([x], rval,
6                       allow_input_downcast = True)
7
8  assert 14 == fn([1, 2, 3])
```

- execution speed optimization
- symbolic differentiation
- stability optimization

Université de Montréal

- The fundamental package for scientific computing in Python
- Linear algebra (and more) on multidimensional tensors
- Random number generators
- and much, much more
- all in the friendly python syntax

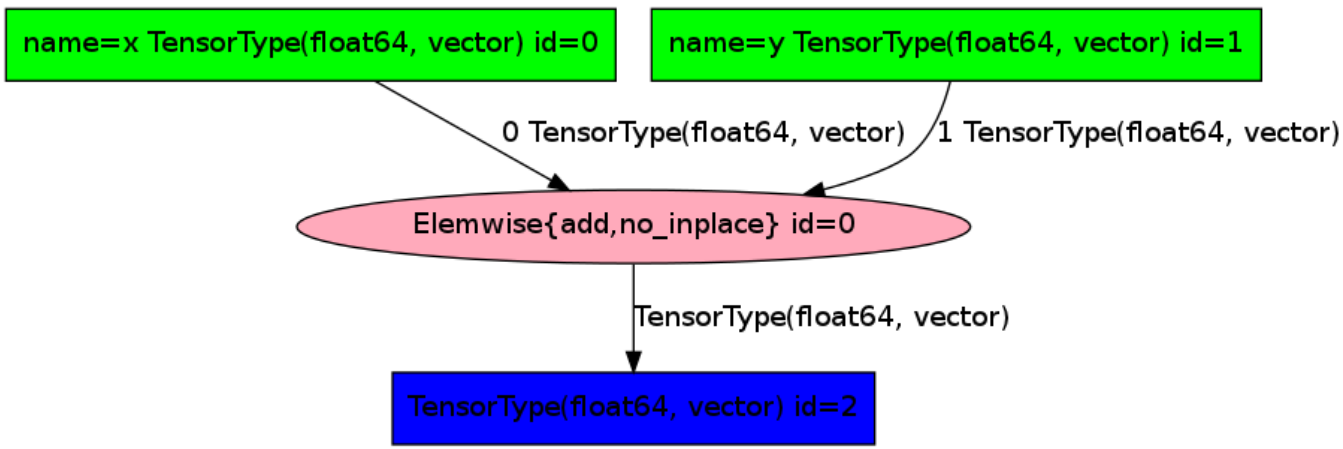
```
1 import numpy
2 W = numpy.random.uniform(size=(3,3))
3 X = numpy.asarray([1,0,1])
4 print numpy.dot(W,X)
```

See ipython notebook example.

Université de Montréal

Is as simple as following these simple steps:

1. Declare *variables*, specifying the types
2. Write down the expression in terms of the variables
3. *Compile the function* that will compute the expression
4. Call the function on the intended data



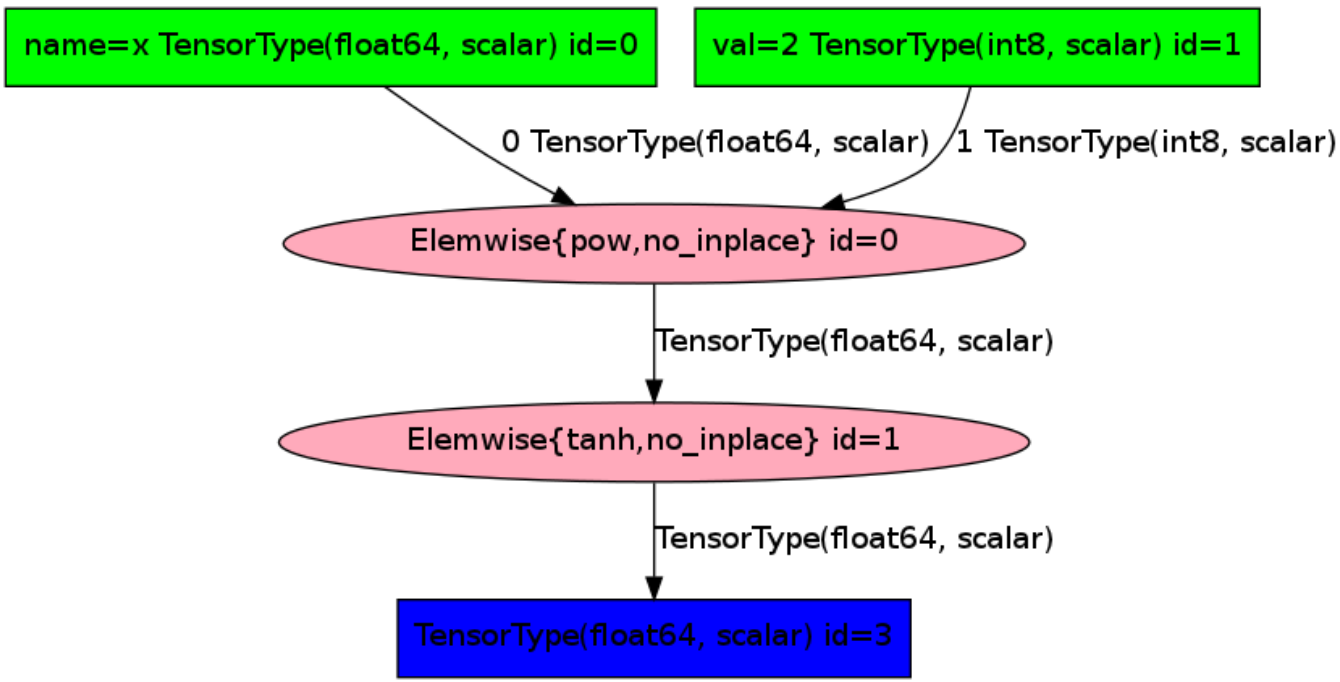
See ipython notebook example.

Université de Montréal

- Theano provides symbolic differentiation
- The computational graph representing the gradients is automatically optimized
- And all this at a single call away: `TT.grad(cost, wrt)`

```
1  import theano, theano.tensor as TT
2  x = TT.vector('x')
3  rval = TT.tanh(x ** 2)
4  fn_forward = theano.function([x], rval)
5  gx = TT.grad(rval, x)
6  fn_grad = theano.function([x], gx)
```

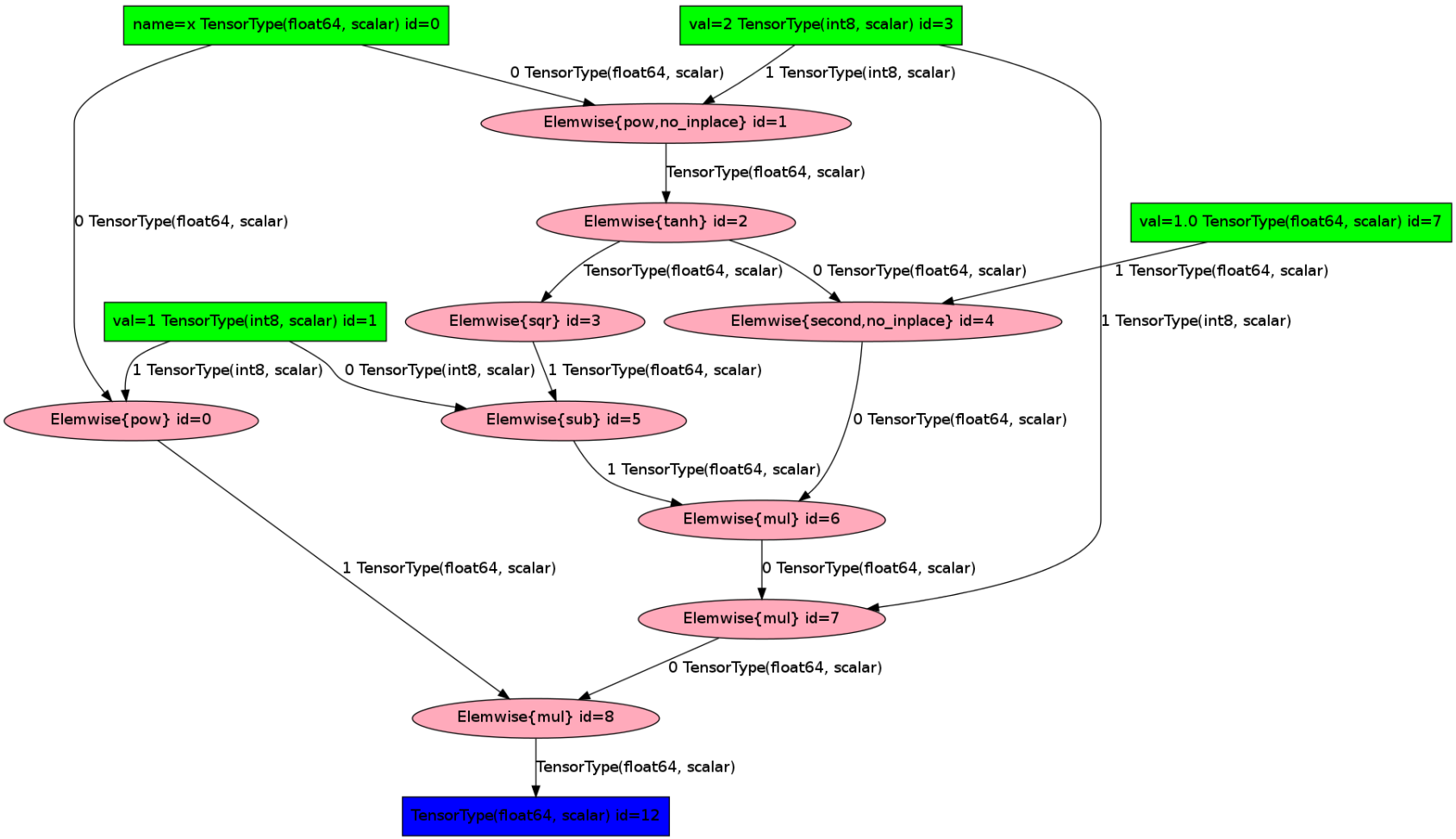
Université de Montréal



$$\frac{\partial \tanh(x^2)}{\partial x} = \frac{\partial \tanh(x^2)}{\partial x^2}\frac{\partial x^2}{\partial x} = \frac{\partial \tanh(x^2)}{\partial \tanh(x^2)}\frac{\partial \tanh(x^2)}{\partial x^2}\frac{\partial x^2}{\partial x} \quad (1)$$

Université de Montréal

Computing derivatives



Université de Montréal

- Think of them as global variables
- They have a state that stays consistent in between calls
- Especially useful to keep memory on the same device
- Can be updated via `set_value` and `updates` provided to `theano.function`

See ipython notebook example.

Université de Montréal

```
1         def function(inputs,
2                      outputs=None,
3                      mode=None,
4                      updates=[],
5                      givens=[],
6                      no_default_updates=False,
7                      accept_inplace=False,
8                      name=None,
9                      rebuild_strict=True,
10                     allow_input_downcast=None,
11                     profile=None):
```

Université de Montréal

- Use **test values**
- Use `theano.printing.Print` to print intermediate results
- Use `theano.printing.pydotprint` to visualize the computational graph
- Run in `DEBUG_MODE`

See ipython notebook example.

Université de Montréal

See ipython notebook example.

http://deeplearning.net/tutorial/logreg.html#logreg

Université de Montréal

Questions ?

Université de Montréal