Theano

Intel-Unesp/NCC - Machine Learning and High Performance Computing Team

Machine Learning Team

June 23, 2017

Overview

- Introduction to Theano
- What is Theano? Is it a programming language?
- Basic Algebra
- Logistic Function in Theano - A Neural Activation Function
- Functions with Multiple Output
- Theano: Using Shared Variables
- Theano: Computing Gradiates
- Theano: Math Challenge
- References

Introduction to Theano

Theano is a Python library that lets you to define, optimize, and evaluate mathematical expressions.

Theano combines aspects of a computer algebra system (CAS) with aspects of an optimizing compiler.

Introduction to Theano

Symbolic Expression

Creating the tensors

```
import theano
from theano import tensor
# these lines define two symbolic floating-point scalars!
a = tensor.dscalar()
b = tensor.dscalar()
...
...
```

Introduction to Theano

Design an expression with the tensors

```
...
#this part create a simple math expression
c = a + b
```

Introduction to Theano

Transforming the expression in a callable code

```
...
...
#this fragment convert the expression (c = a + b)
#into a callable code that take as inputs (a,b) and
#returns c.
f = theano.function([a,b], c)
...
...
```

Introduction to Theano

Calling the object code (evaluating the function with real parameters)

```
...
...
a = 1.5
b = 2.5
#computing c = 4
c = f(a,b)
...
...
```

Theano: Is it a programming language?

When programming in theano you have to:

1. declare variables and to define their types
2. build expressions for how to put those variables together
3. compile expression graphs to functions in order to use them for computation.

So, what is Theano?

Theano: Is it a programming language?

It is good to think of theano as the interface to a compiler which can build a callable object from a purely symbolic graph.

One of Theanos most important features is that theano.function can optimize a graph and compile some or all of it into native machine instructions.

Theano: Basic Algebra

Adding two matrices:

```
import numpy
import theano.tensor as T
from theano import function
...
#The first steps is to instantiate two symbols (variables)
#representing the quantities that you want to add.
...
x = T.dmatrix('x')
y = T.dmatrix('y')
...
z = x + y #build an expression

#use the function on 2D arrays
f = function([x, y], z)
```

Theano: Basic Algebra. *HandsOn!*

Exercise: Run this fragment. What is/are the result(s)?

```
import theano
a = theano.tensor.vector()
out = a + a ** 10
f = theano.function([a], out)
print(f([0, 1, 2]))
```

Theano: *Logistic Function*

Let's compute the Logistic Curve

$$s(x) = \frac{1}{1 + e^{-x}}$$

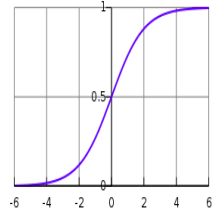

Figure: Logistic Function

Theano Code: *Logistic Function*

```
import theano
import theano.tensor as T
x = T.dmatrix('x')
s = 1 / (1 + T.exp(-x))
logistic = theano.function([x], s)
...
...
#Let's evaluate it ...
logistic([[0, 1], [-1, -2]])
...
#Are the results correct? - Verify!
```

Theano: Functions with Multiple Output

```
import theano
import theano.tensor as T
a, b = T.dmatrices('a','b')
diff = a - b
abs_diff = abs(diff)
diff_squared = diff*2
f = theano.function([a,b],[diff,abs_diff,diff_squared])
...
#Try it using
#f([[1, 1], [1, 1]], [[0, 1], [2, 3]])
```

Theano: Using Shared Variables

It is possible to build functions that memorize their internal states. In this case, at the beginning, the state is zero.Then, on each function call, the state is incremented by the functions argument.

```
from theano import shared
state = shared(0)
inc = T.iscalar('inc')
accumulator = function([inc], state, updates=[(state, state+inc)])
...
#give an real use to this function
...
#Let's try it out!
print(state.get_value())
accumulator(1)
print(state.get_value())
accumulator(300)
print(state.get_value())
```

Theano: Computing Gradients

Let's create a function to compute the derivative of some expression y with respect to its parameter x. $y(x) = x^2$

```
import numpy
import theano
import theano.tensor as T
from theano import pp
x = T.dscalar('x')
y = x ** 2
gy = T.grad(y, x)
pp(gy)
f = theano.function([x], gy)
#...
#...
#Try it out !
f(4)
```

Theano: Math Challenge

$$s(x) = \frac{1}{1+e^{-x}}$$

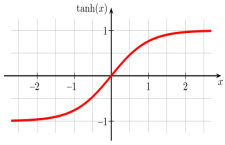

Figure: Logistic and Hyperbolic Tangent Functions

Hands On! - Write a code in Theano to find the Gradient of the Logistic and Hyperbolic Tangent Functions

- Plot the curves of both gradients and find out their outputs at x = -1.5, 0 and 1.5

References


Theano Documentation - http://deeplearning.net/software/theano/


ZOCCA, V.; *et.al.* Python Deep Learning. London:Packt Books, 2017