OpenDeep Documentation

Release 0.0.9a

Vitruvian Science

March 12, 2017

Contents

Developer hub: http://www.opendeep.org/

OpenDeep is a deep learning framework for Python built from the ground up in Theano with a focus on flexibility and ease of use for both industry data scientists and cutting-edge researchers. OpenDeep is a modular and easily extensible framework for constructing any neural network architecture to solve your problem.

Use OpenDeep to:

- Quickly prototype complex networks through a focus on complete modularity and containers similar to Torch.
- Configure and train existing state-of-the-art models.
- Write your own models from scratch in Theano and plug into OpenDeep for easy training and dataset integration.
- Use visualization and debugging tools to see exactly what is happening with your neural net architecture.
- Plug into your existing Numpy/Scipy/Pandas/Scikit-learn pipeline.
- Run on the CPU or GPU.

This library is currently undergoing rapid development and is in its alpha stages.

CHAPTER 1

Quick example usage

Train and evaluate a Multilayer Perceptron (MLP - your generic feedforward neural network for classification) on the MNIST handwritten digit dataset:

```
from opendeep.models import Prototype, Dense, Softmax
from opendeep.models.utils import Noise
from opendeep.optimization.loss import Neg_LL
from opendeep.optimization import AdaDelta
from opendeep.data import MNIST
from theano.tensor import matrix, lvector

print "Getting data..."
data = MNIST()

print "Creating model..."
in_shape = (None, 28*28)
in_var = matrix('xs')
mlp = Prototype()
mlp.add(Dense(inputs=(in_shape, in_var), outputs=512, activation='relu'))
mlp.add(Noise, noise='dropout', noise_level=0.5)
mlp.add(Dense, outputs=512, activation='relu')
mlp.add(Noise, noise='dropout', noise_level=0.5)
mlp.add(Softmax, outputs=10, out_as_probs=False)

print "Training..."
target_var = lvector('ys')
loss = Neg_LL(inputs=mlp.models[-1].p_y_given_x, targets=target_var, one_
↪hot=False)

optimizer = AdaDelta(model=mlp, loss=loss, dataset=data, epochs=10)
optimizer.train()
```

```
print "Predicting..."
predictions = mlp.run(data.test_inputs)

print "Accuracy: ", float(sum(predictions==data.test_targets)) / len(data.
↪test_targets)
```

Congrats, you just:

- set up a dataset (MNIST)
- instantiated a Prototype container model
- added fully-connected (dense) layers and dropout noise to create an MLP
- trained it with an AdaDelta optimizer
- and predicted some outputs given inputs!

CHAPTER 2

Installation

Because OpenDeep is still in alpha, you have to install via setup.py. Also, please make sure you have these dependencies installed first.

Dependencies

- Theano: Theano and its dependencies are required to use OpenDeep. You need to install the bleeding-edge version directly from their GitHub, which has installation instructions here.
  - For GPU integration with Theano, you also need the latest CUDA drivers. Here are instructions for setting up Theano for the GPU. If you prefer to use a server on Amazon Web Services, here are instructions for setting up an EC2 gpu server with Theano.
  - CuDNN (optional but recommended for CNN's): for a fast convolution support from Nvidia. You will want to move the files to Theano's directory like the instructions say here: Theano cuDNN integration.
- Six: Python 2/3 compatibility library.
- Pillow (PIL) (optional): image manipulation functionality.
- PyYAML (optional): used for YAML parsing of config files.
- Bokeh (optional): if you want live charting/plotting of values during training or testing.

Install from source

1. Navigate to your desired installation directory and download the github repository:

```
git clone https://github.com/vitruvianscience/opendeep.git
```

2. Navigate to the top-level folder (should be named OpenDeep and contain the file setup.py) and run setup.py with develop mode:

```
cd opendeep
python setup.py develop
```

Using `python setup.py develop` instead of the normal `python setup.py install` allows you to update the repository files by pulling from git and have the whole package update! No need to reinstall when you get the latest files.

That's it! Now you should be able to import opendeep into python modules.

CHAPTER 3

More Information

Source code: https://github.com/vitruvianscience/opendeep

Documentation and tutorials: http://www.opendeep.org/

User group: opendeep-users

Developer group: opendeep-dev

Twitter: @opendeep

We would love all help to make this the best library possible! Feel free to fork the repository and join the Google groups!

CHAPTER 4

Why OpenDeep?

- **Modularity**. A lot of recent deep learning progress has come from combining multiple models. Existing libraries are either too confusing or not easily extensible enough to perform novel research and also quickly set up existing algorithms at scale. This need for transparency and modularity is the main motivating factor for creating the OpenDeep library, where we hope novel research and industry use can both be easily implemented.
- **Ease of use**. Many libraries require a lot of familiarity with deep learning or their specific package structures. OpenDeep's goal is to be the best-documented deep learning library and have smart enough default code that someone without a background can start training models, while experienced practitioners can easily create and customize their own algorithms.
- **State of the art**. A side effect of modularity and ease of use, OpenDeep aims to maintain state-of-the-art performance as new algorithms and papers get published. As a research library, citing and accrediting those authors and code used is very important to the library.

CHAPTER 5

Contents:

opendeep package

Subpackages

opendeep.data package

Subpackages

opendeep.data.standard_datasets package

Subpackages

opendeep.data.standard_datasets.image package

Submodules

opendeep.data.standard_datasets.image.cifar10 module

opendeep.data.standard_datasets.image.mnist module

Module contents

opendeep.data.standard_datasets.midi package

Submodules

opendeep.data.standard_datasets.midi.jsb_chorales module

opendeep.data.standard_datasets.midi.musedata module

opendeep.data.standard_datasets.midi.nottingham module

opendeep.data.standard_datasets.midi.piano_midi_de module

Module contents

Module contents

opendeep.data.stream package

Subpackages

opendeep.data.stream.tests package

Submodules

opendeep.data.stream.tests.test_filestream module

opendeep.data.stream.tests.test_modifystream module

Module contents

Submodules

opendeep.data.stream.batchstream module

opendeep.data.stream.filestream module

opendeep.data.stream.modifystream module

Module contents

opendeep.data.tests package

Submodules

opendeep.data.tests.test_chars module

opendeep.data.tests.test_cifar10 module

opendeep.data.tests.test_dataset module

opendeep.data.tests.test_memory module

opendeep.data.tests.test_midi module

opendeep.data.tests.test_mnist module

Module contents

Submodules

opendeep.data.dataset module

opendeep.data.dataset_file module

opendeep.data.dataset_image module

opendeep.data.dataset_memory module

opendeep.data.text module

Module contents

opendeep.log package

Subpackages

opendeep.log.tests package

Submodules

opendeep.log.tests.test_logger module

Module contents

Submodules

opendeep.log.logger module

Module contents

opendeep.models package

Subpackages

opendeep.models.container package

Submodules

opendeep.models.container.prototype module

opendeep.models.container.repeating module

Module contents

opendeep.models.single_layer package

Subpackages

opendeep.models.single_layer.tests package

Submodules

opendeep.models.single_layer.tests.basic_mlp_mnist module

opendeep.models.single_layer.tests.lenet module

opendeep.models.single_layer.tests.rbm_mnist module

opendeep.models.single_layer.tests.recurrent module

opendeep.models.single_layer.tests.softmax_mnist module

Module contents

Submodules

Submodules

opendeep.models.model module

Module contents

opendeep.monitor package

Subpackages

opendeep.monitor.tests package

Submodules

opendeep.monitor.tests.monitor_bokeh_server module

opendeep.monitor.tests.test_fileservice module

Module contents

Submodules

opendeep.monitor.monitor module

opendeep.monitor.out_service module

opendeep.monitor.plot module

Module contents

opendeep.optimization package

Subpackages

opendeep.optimization.loss package

Submodules

opendeep.optimization.loss.binary_crossentropy module

opendeep.optimization.loss.categorical_crossentropy module

Submodules

opendeep.utils.midi.DataTypeConverters module

opendeep.utils.midi.EventDispatcher module

opendeep.utils.midi.MidiFileParser module

opendeep.utils.midi.MidiInFile module

opendeep.utils.midi.MidiInStream module

opendeep.utils.midi.MidiOutFile module

opendeep.utils.midi.MidiOutStream module

opendeep.utils.midi.MidiToText module

opendeep.utils.midi.RawInstreamFile module

opendeep.utils.midi.RawOutstreamFile module

opendeep.utils.midi.constants module

opendeep.utils.midi.example_mimimal_type0 module

opendeep.utils.midi.example_print_channel_0 module

opendeep.utils.midi.example_print_events module

opendeep.utils.midi.example_print_file module

opendeep.utils.midi.example_transpose_octave module

opendeep.utils.midi.utils module

Module contents

opendeep.utils.tests package

Submodules

opendeep.utils.tests.test_batch module

Module contents

Submodules

opendeep.utils.activation module

opendeep.utils.batch module

opendeep.utils.config module

opendeep.utils.constructors module

opendeep.utils.conv1d_implementations module

opendeep.utils.decay module

opendeep.utils.decorators module

opendeep.utils.file_ops module

opendeep.utils.image module

opendeep.utils.misc module

opendeep.utils.nnet module

opendeep.utils.noise module

opendeep.utils.regularization module

opendeep.utils.statistics module

Module contents

Submodules

opendeep.version module

Module contents

Indices and tables

- genindex
- modindex
- search