thdl Documentation

Release 0.1.0

Chao-Ming Wang

May 05, 2017

Contents:

TheanoDL, also called **thdl**, is a deep learning library on the top of Theano.

The API design is based on the guidance of software design philosophy.

CHAPTER 1

User Guides

Installation

NumpyDL has a couple of prerequisites that need to be installed first, but it is not very picky about versions. The most important package is Numpy. At the same time, you should install some other useful packages, such as scipy and scikit-learn. Most importantly, these packages are not required to install the specific version to fit the version of NumpyDL you choose to install.

We strongly recommend you to install the Miniconda or a bigger installer Anaconda which is a leading open data science platform powered by Python and well integrated the efficient scientific computing platform MKL.

Prerequisites

Python + pip

NumpyDL currently requires Python 3.3 or higher to run. Please install Python via the package manager of your operating system if it is not included already.

Python includes `pip` for installing additional modules that are not shipped with your operating system, or shipped in an old version, and we will make use of it below. We recommend installing these modules into your home directory via `--user`, or into a virtual environment via `virtualenv`.

C compiler

Numpy/scipy require a C compiler if you install them via `pip`. On Linux, the default compiler is usually‘‘gcc‘‘, and on Mac OS, it’s `clang`. On Windows, we recommend you to install the Miniconda or Anaconda. Again, please install them via the package manager of your operating system.

numpy/scipy + BLAS

NumpyDL requires numpy of version 1.6.2 or above, and sometimes also requires scipy 0.11 or above. Numpy/scipy rely on a BLAS library to provide fast linear algebra routines. They will work fine without one, but a lot slower, so it is worth getting this right (but this is less important if you plan to use a GPU).

If you install numpy and scipy via your operating system's package manager, they should link to the BLAS library installed in your system. If you install numpy and scipy via `pip install numpy` and `pip install scipy`, make sure to have development headers for your BLAS library installed (e.g., the `libopenblas-dev` package on Debian/Ubuntu) while running the installation command. Please refer to the numpy/scipy build instructions if in doubt.

Stable NumpyDL release

NumpyDL 0.1 requires a more recent version of Theano than the one available on PyPI. To install a version that is known to work, run the following command:

```
pip install -r https://github.com/oujago/NumpyDL/blob/master/requirements.txt
```

```
pip install npdl
```

If you do not use `virtualenv`, add `--user` to both commands to install into your home directory instead. To upgrade from an earlier installation, add `--upgrade`.

Development installation

Alternatively, you can install NumpyDL from source, in a way that any changes to your local copy of the source tree take effect without requiring a reinstall. This is often referred to as *editable* or *development* mode. Firstly, you will need to obtain a copy of the source tree:

```
git clone https://github.com/oujago/NumpyDL.git
```

It will be cloned to a subdirectory called `NumpyDL`. Make sure to place it in some permanent location, as for an *editable* installation, Python will import the module directly from this directory and not copy over the files. Enter the directory and install the known good version of Theano:

```
cd NumpyDL
pip install -r requirements.txt
```

To install the NumpyDL package itself, in editable mode, run:

```
pip install --editable
```

As always, add `--user` to install it to your home directory instead.

Optional: If you plan to contribute to NumpyDL, you will need to fork the NumpyDL repository on GitHub. This will create a repository under your user account. Update your local clone to refer to the official repository as `upstream`, and your personal fork as `origin`:

```
git remote rename origin upstream
git remote add origin https://github.com/<your-github-name>/NumpyDL.git
```

If you set up an SSH key, use the SSH clone URL instead: `git@github.com:<your-github-name>/NumpyDL.git`.

You can now use this installation to develop features and send us pull requests on GitHub, see development!

CHAPTER 2

API References

thdl.utils

thdl.utils.common

thdl.utils.common.**now**()
Get the format time of NOW.

For example:

```
>>> now()
>>> "2017-04-26-16-44-56"
```

Returns `str` instance.

thdl.utils.common.**today**()
Get the format time of TODAY.

For example:

```
>>> today()
>>> "2017-04-26"
```

Returns `str` instance.

thdl.utils.common.**time_format**(*total_time*)
Change the total time into the normal time format.

For examples:

```
>>> time_format(36)
>>> "36 s"
>>> time_format(90)
```

```
>>> "1 min 30 s "
>>> time_format(5420)
>>> "1 h 30 min 20 s"
>>> time_format(20.5)
>>> "20 s 500 ms"
```

Parameters **total_time** – float or str instance. The total seconds of the time.

Returns str instance. The format string about time.

thdl.utils.common.**dict_to_str**(*dict_obj*, *js='-'*)
Change dict object to string.

Parameters

- **dict_obj** – dict instance. The dict object to string.
- **js** – str instance. The join symbol of keys and values.

Returns str instance. Return the string object.

thdl.utils.file

thdl.utils.file.**write_xls**(*contents*, *filepath*)
Write the contents into the xls tables.

Parameters

- **contents** – an instance of dict or an instance of list. If *isinstance(contents, dict) == True*, there is only one line in xls table. If contents is a list, then there are several lines in xls table.
- **filepath** – str instance. The path to save.

thdl.utils.file.**pickle_dump**(*data*, *path*)
Given the DATA, then pickle it into the PATH.

Parameters

- **data** – the f_data to pickle.
- **path** – the path to store the pickled f_data.

thdl.utils.file.**pickle_load**(*path*)
From the PATH get the DATA.

Parameters **path** – the path that store the pickled f_data

thdl.utils.file.**check_duplicate_path**(*filepath*)
Check the duplicate path.

If there is a same filepath, for example 'file/tests.txt', Then this path will be 'file/tests(1).txt'

If 'file/tests(1).txt' also exists, then this path will be 'file/tests(2).txt'

Parameters **filepath** – the file path to save.

Returns the file path can be used to save path.

thdl.utils.common

now	Get the format time of NOW.

Continued on next page

Table 2.1 – continued from previous page

today	Get the format time of TODAY.
time_format	Change the total time into the normal time format.
dict_to_str	Change dict object to string.

thdl.utils.file

write_xls	Write the contents into the xls tables.
pickle_dump	Given the DATA, then pickle it into the PATH.
pickle_load	From the PATH get the DATA.
check_duplicate_path	Check the duplicate path.

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

Python Module Index

t

Index