pyKE Documentation

Release 0.1

Johannes Ruthmann, Jan-Christoph Kalo

Jul 14, 2018

Contents:

1 README **3**
- 1.1 Overview . . . 3
- 1.2 Installation . . . 3
- 1.3 Quickstart . . . 3
- 1.4 Interfaces . . . 4
- 1.5 Notes . . . 5

2 API reference **7**
- 2.1 pyke package . . . 8

3 Indices and tables **13**

Python Module Index **15**

Welcome to pyKE's documentation. In this documentation you can find all information about the project.

CHAPTER 1

README

An Open-source library for Knowledge Embedding forked from github.org/thunlp/OpenKE. The original API changed drastically to be more pythonic.

1.1 Overview

This is an implementation based on [TensorFlow](http://www.tensorflow.org) for knowledge representation learning (KRL). It includes native C++ implementations for underlying operations such as data preprocessing and negative sampling. For each specific model, it is implemented by TensorFlow with Python interfaces so that there is a convenient platform to run models on GPUs.

1.2 Installation

1. Clone repository and enter directory

```
git clone https://github.com/ifis-tu-bs/pyKE.git
cd pyKE
```

2. Install package

```
python setup.py install
```

1.3 Quickstart

To compute a knowledge graph embedding, first import datasets and set configure parameters for training, then train models and export results. Here is an example to train the FB15K dataset with the TransE model.

```
from pyke.dataset import Dataset
from pyke.embedding import Embedding
from pyke.models import TransE

# Read the dataset
dataset = Dataset("./benchmarks/fb15k.nt")
embedding = Embedding(
    dataset,
    TransE,
    folds=20,
    epochs=20,
    neg_ent=1,
    neg_rel=0,
    bern=False,
    workers=4,
    dimension=50,  # TransE-specific
    margin=1.0,  # TransE-specific
)

# Train the model. It is saved in the process.
embedding.train(prefix="./TransE", post_epoch=print)

# Save the embedding to a JSON file
embedding.save_to_json("TransE.json")
```

1.4 Interfaces

The class *pyke.embedding.Embedding* represents an embedding which requires a dataset and a model class. Initialize your data set in form of a N-triples file with the class *pyke.dataset.Dataset*.

1.4.1 Models

The class *pyke.models.base.BaseModel* declares the methods that all implemented model classes share, including the loss function neccessairy for training (inserting information into the model) and prediction (aka. retrieving information from the model). This project implements the following model classes:

- RESCAL
- TransE
- TransH
- TransR
- TransD
- HolE
- ComplEx
- DistMult

1.5 Notes

The original fork consists of a C++ library which is compiled once you use the project. Please note, that the compilation is only supported on **UNIX-based systems**. In the future the C++ library should be replaced by a python library.

CHAPTER 2

API reference

2.1 pyke package

2.1.1 Subpackages

pyke.models package

Submodules

pyke.models.ComplEx module

pyke.models.DistMult module

pyke.models.HolE module

pyke.models.RESCAL module

pyke.models.TransD module

pyke.models.TransE module

pyke.models.TransH module

pyke.models.TransR module

pyke.models.base module

class pyke.models.base.**BaseModel**(*ent_count=None*, *rel_count=None*, *batch_size=0*, *variants=0*, *optimizer=None*, *norm_func=<function l1>*, *per_process_gpu_memory_fraction=0.5*)

Bases: object

Properties and behaviour that different embedding models share.

entity(*head=None*)
Embeds a batch of subjects.

fit(*head*, *tail*, *label*, *score*)
Trains the model on a batch of weighted statements.

get_all_instance(*in_batch=False*)

get_all_labels(*in_batch=False*)

get_negative_instance(*in_batch=True*)

get_positive_instance(*in_batch=True*)

get_predict_instance()

predict(*head*, *tail*, *label*)
Evaluates the model's scores on a batch of statements.

relation(*label=None*)
Embeds a batch of predicates.

restore(*prefix: str*)
Reads a model from filesystem.

Parameters **prefix** – Model prefix of the model to laod

save(*prefix: str*, *step: int = None*)
Save the model to filesystem.

Parameters

- **prefix** – File prefix for the model
- **step** – Step of the model (appended to prefix)

save_to_json(*filename: str*)
Save the embedding as JSON file. The JSON file contains the embedding parameters (e.g. entity and relation matrices). These parameters depend on the model.

Parameters **filename** – Filename for the output JSON file

Module contents

2.1.2 Submodules

2.1.3 pyke.dataset module

2.1.4 pyke.embedding module

2.1.5 pyke.library module

class pyke.library.**Library**
Bases: object

Manages the connection to the library.

CPP_BASE = 'cpp_library/Base.cpp'

MAKE_SCRIPT = 'cpp_library/make.sh'

static compile_library(*destination: str*)
Compile the library to the path `destination`.

Parameters **destination** – path for the library

static get_library(*temp_dir: str = None*, *library_name: str = None*)
Return the C++ library. The function compiles it if it doesn't exist and it loads the library.

Parameters

- **temp_dir** – directory where the library is saved (optional)
- **library_name** – filename of the library

Returns c++ library

library = None

library_name = 'pyke.so'

static load_library(*path: str*)
Loads the library from *path*.

Parameters **path** – path to the library (.so)

temp_dir = '.pyke'

2.1.6 pyke.norm module

pyke.norm.**l1**(*vectors*)
Implements the l1 norm on a vectorspace.

Parameters vectors - Tensor of dimension at least one, returning vectors whose norm is to be computed.

Return Value Tensor of reduced dimension returning the norms. The order is preserved.

pyke.norm.**l2**(*vectors*)
Implements the euclidean norm on a vectorspace.

Parameters vectors - Tensor of dimension at least one, returning vectors whose norm is to be computed.

Return Value Tensor of reduced dimension returning the norms. The order is preserved.

2.1.7 pyke.parser module

2.1.8 pyke.utils module

pyke.utils.**get_array_pointer**(*a*)
Returns the address of the numpy array.

Parameters **a** – Numpy array

Returns Memory address of the array

pyke.utils.**md5**(*filename: str*)
Returns the MD5-hashsum of a file.

Parameters **filename** – Filename

Returns MD5-hashsum of the file

pyke.utils.**split_nt_line**(*line: str*)
Splits a line from a N-triples file into subject, predicate and object.

Parameters `line` – Line from a N-triples file

Returns tuple with subject, predicate, object

2.1.9 Module contents

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

Python Module Index

p

pyke, 11
pyke.library, 9
pyke.models.base, 8
pyke.norm, 10
pyke.utils, 10

Index

B

BaseModel (class in pyke.models.base), 8

C

compile_library() (pyke.library.Library static method), 9
CPP_BASE (pyke.library.Library attribute), 9

E

entity() (pyke.models.base.BaseModel method), 9

F

fit() (pyke.models.base.BaseModel method), 9

G

get_all_instance() (pyke.models.base.BaseModel method), 9
get_all_labels() (pyke.models.base.BaseModel method), 9
get_array_pointer() (in module pyke.utils), 10
get_library() (pyke.library.Library static method), 10
get_negative_instance() (pyke.models.base.BaseModel method), 9
get_positive_instance() (pyke.models.base.BaseModel method), 9
get_predict_instance() (pyke.models.base.BaseModel method), 9

L

l1() (in module pyke.norm), 10
l2() (in module pyke.norm), 10
Library (class in pyke.library), 9
library (pyke.library.Library attribute), 10
library_name (pyke.library.Library attribute), 10
load_library() (pyke.library.Library static method), 10

M

MAKE_SCRIPT (pyke.library.Library attribute), 9
md5() (in module pyke.utils), 10

P

predict() (pyke.models.base.BaseModel method), 9
pyke (module), 11
pyke.library (module), 9
pyke.models.base (module), 8
pyke.norm (module), 10
pyke.utils (module), 10

R

relation() (pyke.models.base.BaseModel method), 9
restore() (pyke.models.base.BaseModel method), 9

S

save() (pyke.models.base.BaseModel method), 9
save_to_json() (pyke.models.base.BaseModel method), 9
split_nt_line() (in module pyke.utils), 10

T

temp_dir (pyke.library.Library attribute), 10