Lerot Documentation

Release stable

May 13, 2015

Contents

1 Prerequisites 3

2 Installation 5

3 Running experiments 7

4 Data 9

5 Extensions 11

6 License 13

This project is designed to run experiments on online learning to rank methods for information retrieval. Below is a short summary of its prerequisites, how to run an experiment, and possible extensions.

CHAPTER 1

Prerequisites

- Python (2.7 or higher)
- PyYaml
- Numpy
- Scipy
- Celery
- Gurobi

(all prerequisites are included in the academic distribution of Enthought Python, e.g., version 7.1)

CHAPTER 2

Installation

Install the prerequisites plus Lerot as follows:

```
$ pip install PyYAML numpy scipy celery
$ git clone https://bitbucket.org/ilps/lerot.git
$ cd lerot
$ python setup.py install
```

CHAPTER 3

Running experiments

1. prepare data in svmlight format, e.g., download the *MQ2007* (see next section on *Data*)

```
$ mkdir data
$ wget http://research.microsoft.com/en-us/um/beijing/projects/letor/LETOR4.0/Data/MQ2007.rar -C
$ unrar x data/MQ2007.rar data/
```

2. prepare a configuration file in yml format, e.g., starting from the template below, store as `config/experiment.yml` (or simply use `config/config.yml` instead)

```
training_queries: data/MQ2007/Fold1/train.txt
test_queries: data/MQ2007/Fold1/test.txt
feature_count: 46
num_runs: 1
num_queries: 10
query_sampling_method: random
output_dir: outdir
output_prefix: Fold1
user_model: environment.CascadeUserModel
user_model_args:
    --p_click 0:0.0,1:0.5,2:1.0
    --p_stop 0:0.0,1:0.0,2:0.0
system: retrieval_system.ListwiseLearningSystem
system_args:
    --init_weights random
    --sample_weights sample_unit_sphere
    --comparison comparison.ProbabilisticInterleave
    --delta 0.1
    --alpha 0.01
    --ranker ranker.ProbabilisticRankingFunction
    --ranker_arg 3
    --ranker_tie random
evaluation:
    - evaluation.NdcgEval
```

3. run the experiment using python:

```
$ python src/scripts/learning-experiment.py -f config/experiment.yml
```

4. summarize experiment outcomes:

```
$ python src/scripts/summarize-learning-experiment.py --fold_dirs outdir
```

   Arbitrarily many folds can be listed per experiments. Results are aggregated over runs and folds. The output format is a simple text file that can be further processed using e.g., gnuplot. The columns are: mean_offline_perf stddev_offline_perf mean_online_perf stddev_online_perf

Data

You can download learning to rank data sets here:

- **GOV**: http://research.microsoft.com/en-us/um/beijing/projects/letor/LETOR3.0/Gov.rar (you'll need files in QueryLevelNorm)
- **OHSUMED**: http://research.microsoft.com/en-us/um/beijing/projects/letor/LETOR3.0/OHSUMED.zip
- **MQ2007**: http://research.microsoft.com/en-us/um/beijing/projects/letor/LETOR4.0/Data/MQ2007.rar (files for supervised learning)
- **MQ2008**: http://research.microsoft.com/en-us/um/beijing/projects/letor/LETOR4.0/Data/MQ2008.rar (files for supervised learning)
- **Yahoo!**: http://webscope.sandbox.yahoo.com/catalog.php?datatype=c
- **MSLR-WEB10K**: http://research.microsoft.com/en-us/um/beijing/projects/mslr/data/MSLR-WEB10K.zip
- **MSLR-WEB30K**: http://research.microsoft.com/en-us/um/beijing/projects/mslr/data/MSLR-WEB30K.zip

Note that Lerot reads from both plain text and .gz files.

CHAPTER 5

Extensions

The code is intended to be extended with new learning and/or feedback mechanisms for future experiments. The most obvious points for extension are:

1. comparison - extend ComparisonMethod to add new interleaving or inference methods; existing methods include balanced interleave, team draft, and probabilistic interleave.
2. retrieval_system - extend OnlineLearningSystem to add a new mechanism for learning from click feedback. New implementations need to be able to provide a ranked list for a given query, and ranking solutions should have the form of a vector.

CHAPTER 6

License