MyMediaLite: A Free Recommender System Library

Zeno Gantner, Steffen Rendle, Christoph Freudenthaler, Lars Schmidt-Thieme
gantner@ismll.de, steffen.rendle@uni-konstanz.de, freudenthaler@ismll.de, schmidt-thieme@ismll.de

Universität Hildesheim

Universität Konstanz

Overview

MyMediaLite addresses the two most common tasks in collaborative filtering: **rating prediction** (e.g. on a scale of 1 to 5 stars) and **item prediction from implicit feedback** (e.g. from clicks or purchase actions).

The library contains implementations of **state-of-the-art methods** for both tasks, as well as **efficient data structures**, evaluation routines, and tools that support the development, deployment, and operation of recommender systems.

Features

Choice:

- dozens of different recommendation methods,
- methods can use collaborative, content, or relational data.

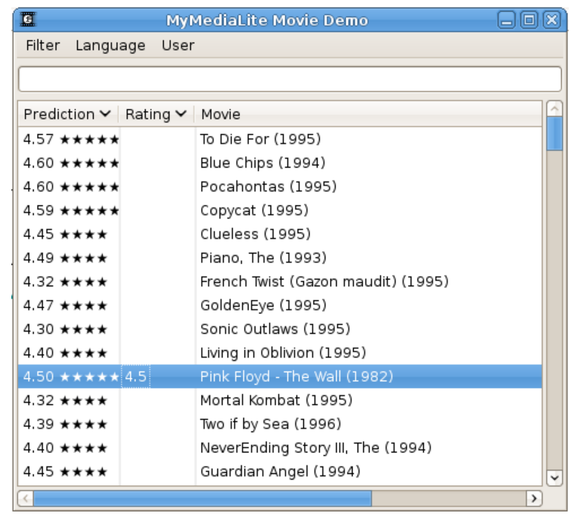
Accessibility:

- includes evaluation routines for rating and item prediction;
- command line tools that read a simple text-based input format;
- usable from C#, Python, Ruby, F#, etc.,
- complete documentation of the library and its tools.

Compactness: Core library is less than 200 KB "big".
Portability: written in C#, for the .NET platform; runs on every architecture where Mono works: Linux, Windows, Mac OS X.
Parallel processing on multi-core/multi-processor systems.
Serialization: save and reload recommender models.
Real-time incremental updates for most recommenders.
Freedom: free/open source software – GNU GPL.

GUI Demo

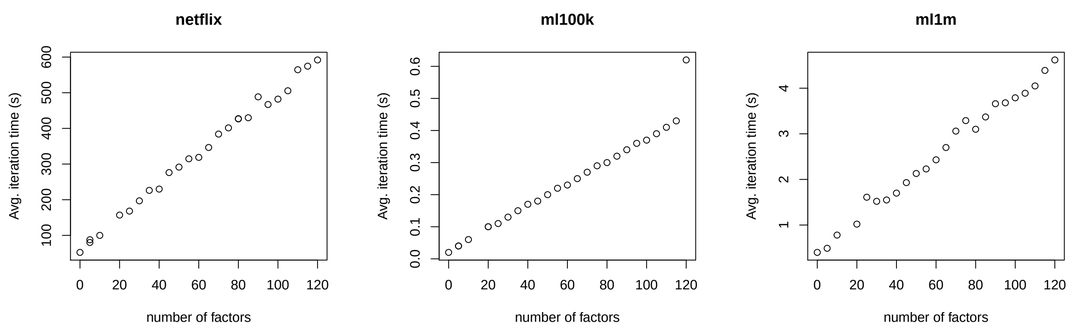
MyMediaLite includes a simple GUI demo that lets the user rate movies.



Using MyMediaLite from Python

```python
#!/usr/bin/env ipy

import clr
clr.AddReference("MyMediaLite.dll")
from MyMediaLite import *

# load the data
user_mapping = Data.EntityMapping()
item_mapping = Data.EntityMapping()
train_data = IO.ItemData.Read("u1.base", user_mapping, item_mapping)
test_users = train_data.AllUsers;
candidate_items = train_data.AllItems;
test_data = IO.ItemData.Read("u1.test", user_mapping, item_mapping)

# set up the recommender
recommender = ItemRecommendation.UserKNN()
recommender.K = 20
recommender.Feedback = train_data
recommender.Train()

# measure the accuracy on the test data set
print Eval.Items.Evaluate(recommender, test_data, train_data, test_users, candidate_items)

# make a prediction for a certain user and item
print recommender.Predict(user_mapping.ToInternalID(1), item_mapping.ToInternalID(1))
```

Performance



Predictive accuracy of several methods on different datasets can be found here: `http://ismll.de/mymedialite/examples/datasets.html`

Methods

Rating Prediction

- averages: global, user, item
- linear baseline by Koren and Bell
- frequency-weighted Slope One (2 variants)
- k-nearest neighbor (kNN):
  - user/item similarities with different similarity measures
  - collaborative or content-based
- matrix factorization: factor-wise/SGD training; RMSE and MAE optimization
- SocialMF (Jamali and Ester)
- parallel SGD for MF (Gemulla et al.)

Item Prediction

- random; most popular item
- SVM using item attributes
- k-nearest neighbor (kNN)
- WR-MF: weighted regularized matrix factorization
- BPR-MF: matrix factorization optimized for Bayesian Personalized Ranking
- BPR-Linear: linear content-based model optimized for BPR

Evaluation

- splits/protocols: user-defined, random split, CV, online
- quality measures: (N)MAE, RMSE, AUC, prec@N, recall@N, MAP, NDCG, MRR
- rating prediction, item prediction (also for groups of users; filtered by attributes, etc.)

Implementing a Recommender Algorithm

```
public class SimpleMatrixFactorization : RatingPredictor, IIterativeModel
{
	Matrix<double> user_factors;
	Matrix<double> item_factors;

	public uint NumFactors { get; set;}
	public double LearnRate { get; set; }
	public virtual double Regularization { get; set; }
	public uint NumIter { get; set; }

	public override void Train()
	{
		// init factor matrices
		user_factors = new Matrix<double>(MaxUserID + 1, NumFactors);
		item_factors = new Matrix<double>(MaxItemID + 1, NumFactors);
		MatrixUtils.InitNormal(user_factors, 0, 0.1);
		MatrixUtils.InitNormal(item_factors, 0, 0.1);

		// learn model parameters
		for (uint current_iter = 0; current_iter < NumIter; current_iter++)
			Iterate();
	}

	public void Iterate()
	{
		foreach (int index in ratings.RandomIndex)
		{
			int u = ratings.Users[index];
			int i = ratings.Items[index];

			double p = Predict(u, i);
			double err = ratings[index] - p;

			for (int f = 0; f < NumFactors; f++)
			{
				double u_f = user_factors[u, f];
				double i_f = item_factors[i, f];
				MatrixUtils.Inc(user_factors, u, f, LearnRate * (err * i_f - Regularization * u_f));
				MatrixUtils.Inc(item_factors, i, f, LearnRate * (err * u_f - Regularization * i_f));
			}
		}
	}

	public override double Predict(int user_id, int item_id)
	{
		return MatrixUtils.RowScalarProduct(user_factors, user_id, item_factors, item_id);
	}
}
```

Related Work

Other recommender system frameworks:

- Java: Apache Mahout, LensKit
- C++: GraphLab, Waffles
- R recommenderlab
- Python: python-recsys, Crab

... more on our website.

Future Plans

- community project
- even simpler usage
- additional recommendation tasks and input types
- REST webservice API (co-operation with the *LensKit* project)
- GUI frontend

Acknowledgements

- co-funded by the European Commission FP7 project MyMedia.
- Chris Newell (BBC R+D) for valuable feedback, and for porting parts of the library to Java.
- all contributors to MyMediaLite (listed in the `README` file).

SEVENTH FRAMEWORK PROGRAMME

Try MyMediaLite

http://ismll.de/mymedialite

MyMedia