# Technical Documentations

Alexander Oberegger

May 2016

## 1 Introduction

This Section provides a short technical introduction into the code written for the WPOI algorithm. The basis of the implementation is the work of Zeno Gantner that resulted in the MyMediaLite [1] project. The code from the WPOI project has to be integrated into the MyMediaLite data structure to work properly. The following steps have to be done to get the project to work:

1. Download the sql database dump from `https://github.com/aoberegg/WPOI/tree/master/database/final_database.rar` and create a database out of it.
2. Checkout the MyMediaLite code [2].
3. Checkout the WPOI code [3] and integrate it into the data structure of MyMediaLite (some of the files already exist, please overwrite).
4. Download Xamarin Studio[4] to work with the project.

### Structure

The application is separated into two parts, namely the MyMediaLite dll that can be found at

`src/MyMediaLite/bin/Release`

and the test program that is located at

`examples/csharp/TestMediaLite/rating_prediction`

[1] `https://github.com/zenogantner/MyMediaLite`
[2] `https://github.com/zenogantner/MyMediaLite`
[3] `https://github.com/aoberegg/WPOI/`
[4] `https://www.xamarin.com/download`

### MyMediaLite DLL

To change the MyMediaLite DLL the project storet at

"src/MyMediaLite/MyMediaLite.sln"

has to be opened in Xamarin Studio. The WPOI algorithm is then implemented in

"src/MyMediatLite/RatingPrediction/WeatherContextAwareItemRecommender"

Please be aware that the weather data is retrieved from the database stored in the MySql dump.

### WPOI Project

The WPOI test project is based on the MyMediaLite DLL and is stored at

"examples/csharp/TestMediaLite/TestMediaLite.sln"

The test programm is implemented in

"examples/csharp/TestMediaLite/rating_prediction.cs"

and provides options for testing the baseline algorithms as well as options for testing WPOI. the resulting executable stored at

"examples/csharp/TestMediaLite/bin/2/Release/TestMediaLite.exe"

can be executed with the following command line options:

```
TestMediaLite.exe "path/to/check-in-data/filename.data"
"SERVER=localhost;DATABASE=<database_name>;UID=<user_id>;
PASSWORD=<password>"
"<algorithm_id>"
"<city_id>"
"<max_iterations>"
"<weather_feature_in_database>"
```

A start of the program with the WPOI algorithm on city 53 with 7000 iterations and the temperature weather feature on Linux OS could look like follows:

```
mono TestMediaLite.exe
"/home/aoberegger/data/evaluation_protocol/city53.data"
"SERVER=localhost;DATABASE=localhost;UID=root;PASSWORD=rootpw"
"4" "53" "7000" "hw.temperature"
```

Table 1 shows the command line options for the TestMediaLite executable.

| **Option** | **Possible values** |
|---|---|
| "path/to/check-in-data/filename.data" | Path to the check-in data file containing ¡user_id, item_id, Timestamp¿ triples. |
| "SERVER=localhost;-DATABASE=localhost-;UID=root-;PASSWORD=rootpw" | The Database credentials. |
| city_id | Every city id available in the database but the id has to fit with the datafile containing the check-ins. |
| max_iterations | The maximum amount of iterations for learning the latent model parameters. |
| weather_feature_in_database | 1 = Beta tuning, 2 = Rank-GeoFM, 3 = Old version of WPOI, 4 = WPOI, 5 = MP, 6 = Item-KNN, 7 = User-KNN, 8 = WRMF, 9 = BPRMF |

Table 1: Command line options for the TestMediaLite executable.