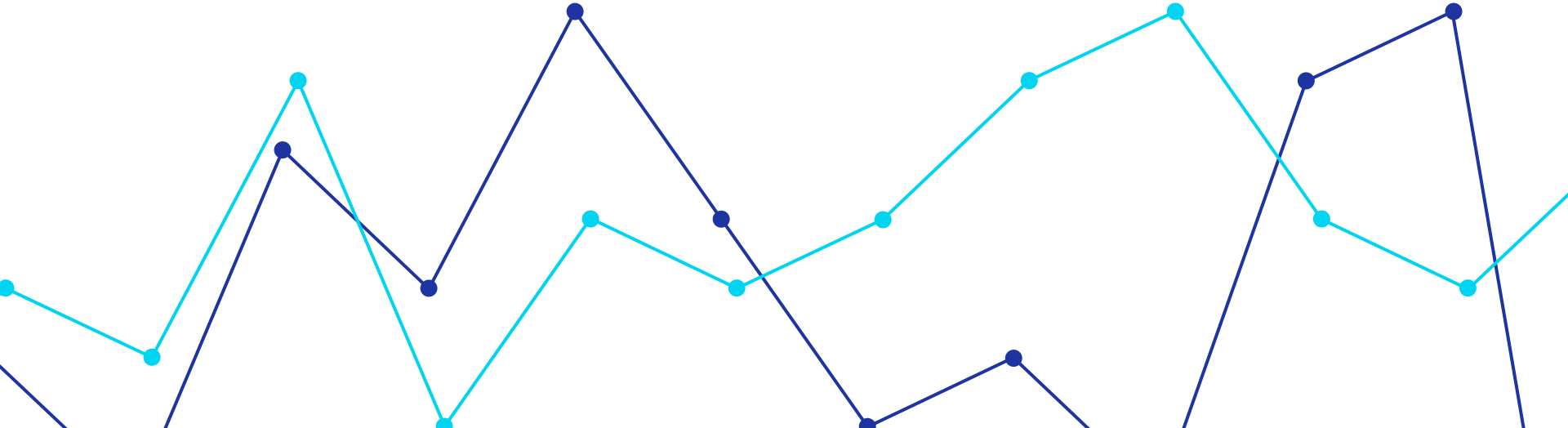
Recommender Systems Challenge

Arianna Galzerano, Francesco Fulco Gonzales



Project Timeline

Data exploration

Analysis of URM and ICM data

Models exploration and evaluation

Hyperparameters optimization and base models comparison

Hybrids

Implementation of different hybrid models starting from the best base models

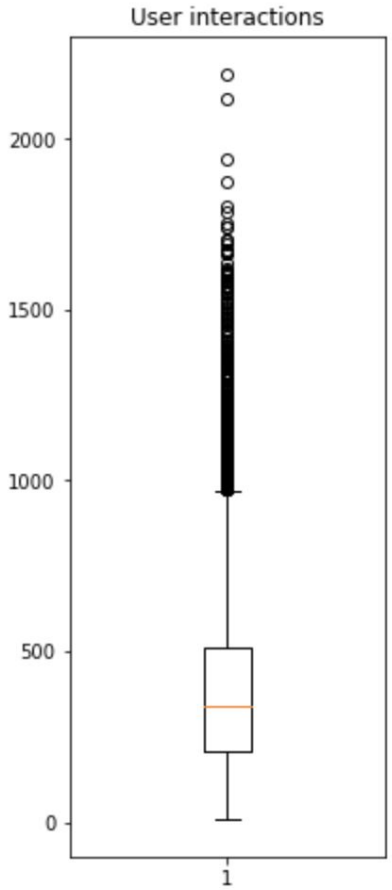
Final model

Structure of the final hybrid model

Cross-validation & other techniques

Techniques to improve performance and avoid overfitting

Understanding the data



- Clean dataset: no missing values or improper data
- All users have at least tens of interactions: CBF not needed

URM

5.29M Interactions

13650 Users

>10 I4U

ICM

7 genres

12 subgenres

112 channels

358K events

Users

N. of interactions in URM:
min= 9, max 2191

ICMS:

Genre, subgenre, channels, event

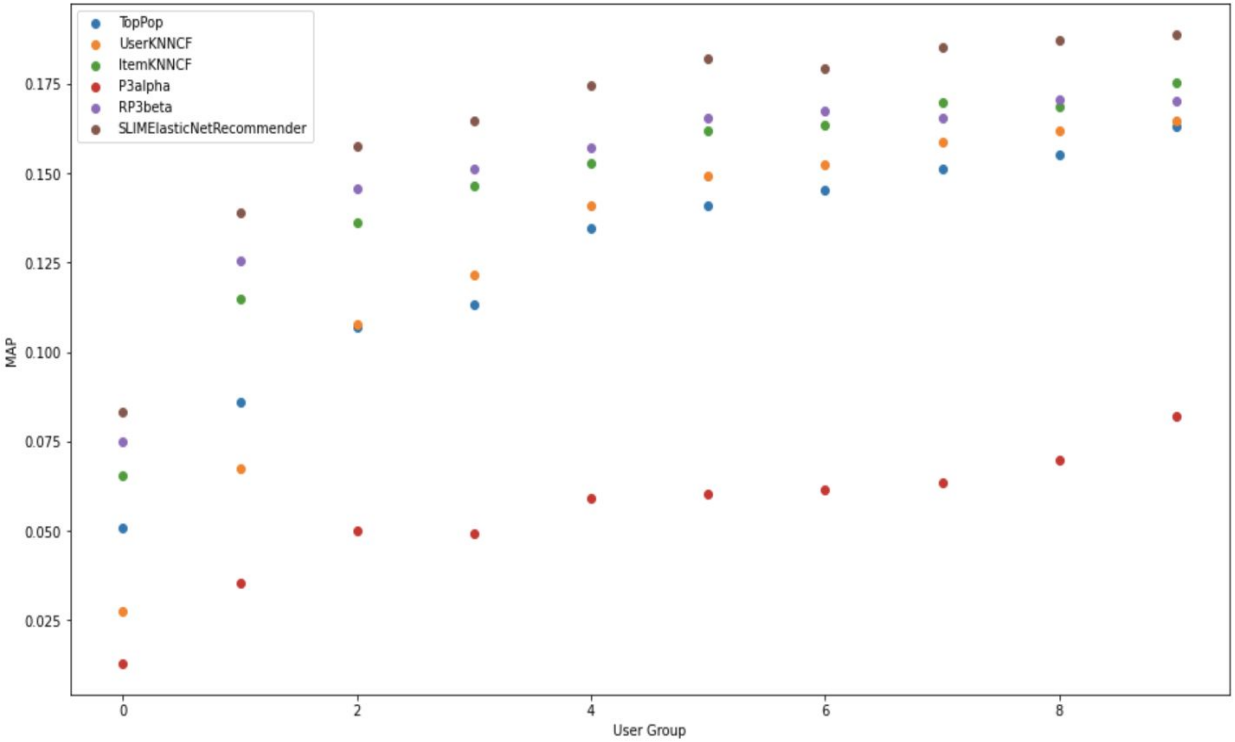
Base Models Evaluation

Base Model	***Valid MAP@10***
SLIMElasticNet	0.2391579
EASE-R	0.2393846
IALS implicit	0.2324677
PureSVD	0.2309224
RP3beta	0.2250362
ItemKNNCF	0.2131692
SLIMBPR	0.2033459

Some techniques

- Stacking URM - ICM
- Train validation split: 80:20 (initially 90:10)
- Bayesian Optimization from course repo
- Change of parameters range for different models during optimization and search
- Training both locally and in the cloud

Preprocessing



User Grouping

We tried segmentation into 2 to 10 groups of users based on n. Of interactions

Stacking

For some of the base models it was useful for performance improvement (EASE-R and Rp3beta)

Hybrids techniques and implementation

Linear Combination

Weighted sum of scores
HYBRID RATINGS class

Similarity Merge

Weighted sum of similarity matrices
HYBRID SIMILARITY CLASS

List Combination

Combination of the recommendation lists of the submodels of the hybrid

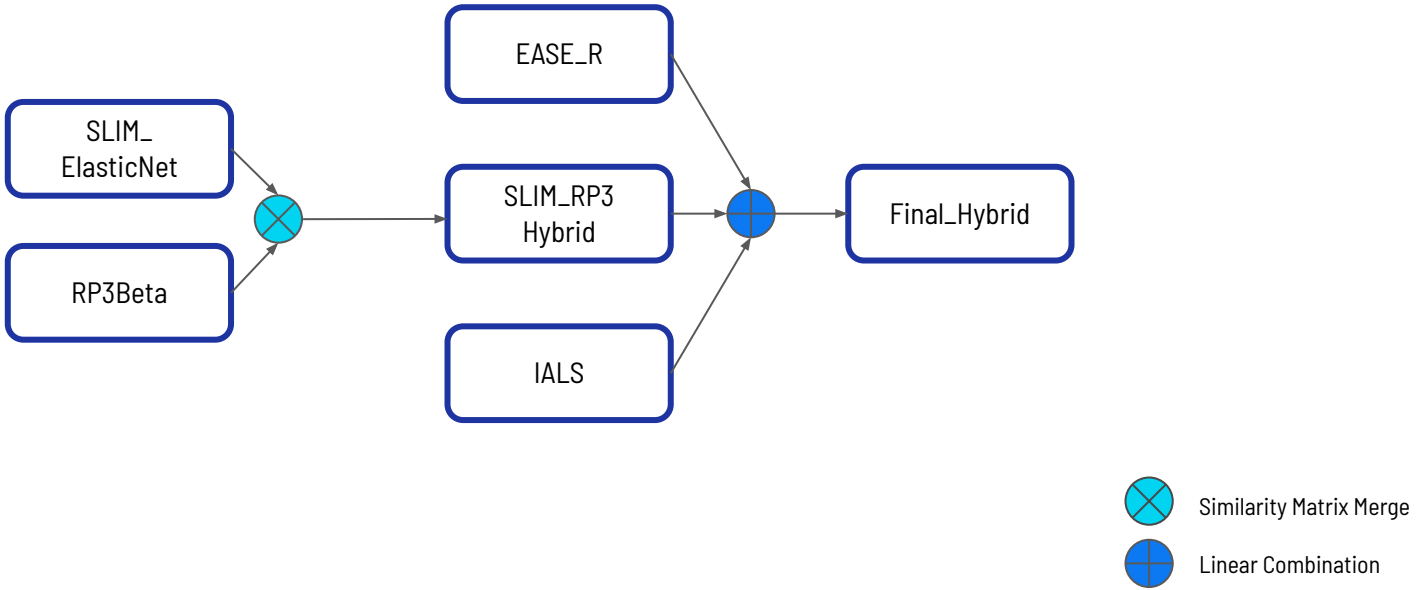
Cotraining for optimization

Joint hyperparameter optimization of models

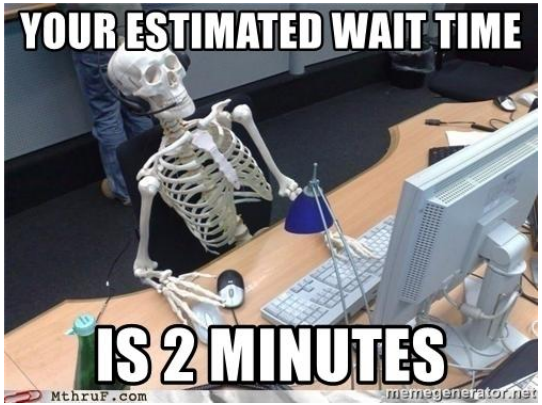
Final Hybrid Model

Private leaderboard MAP score: **0.48656** (7th place)

Stacking URM with:
- ICM channel for SLIM
- ICM event for EASE_R



Other techniques



k-fold cross validation

- k=5 to keep 80:20 split
- Bayesian hyperparameter tuning on cross-validated MAP
- Robustness check for private leaderboard

Selective Cotraining

- Fixed some of the most compute-intensive models to finetune the others

IALS from implicit library

- Significant speed up in training time
- Leveraging Cython and multithreading


Thank you for the attention!
Any questions?
Code and optimization results available on github:
@fulcus & @arigalzi