A comprehensive review of Collaborative Filtering models

From Matrix Factorizations to Graph Neural Networks

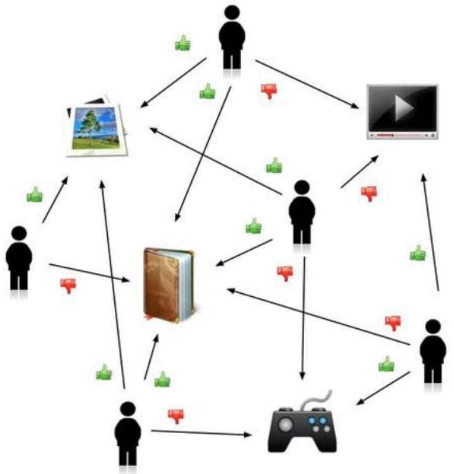
Konstantin Shlychkov
Alexander Kharitonov
Gleb Mazanov
Nikolay Kotoyants
Danil Gusak

Skoltech

Problem, Approaches and Hypothesis

We propose implementation and comparison for two different approaches to solve Collaborative Filtering task:

1. **Matrix Factorization methods**
    - ALS – Alternating Least Squares
    - eALS – element-wise ALS
    - iALS – implicit ALS
2. **Graph Neural Network for Collaborative Filtering**
    - NGCF – Neural Graph Collaborative Filtering



Expectation:

- Better performance with NGCF compared to MF models
- In terms of computational speed — eALS or iALS as the fastest algorithms

Setup

Datasets:

movielens

yelp

Frameworks and technologies:

NVIDIA CUDA

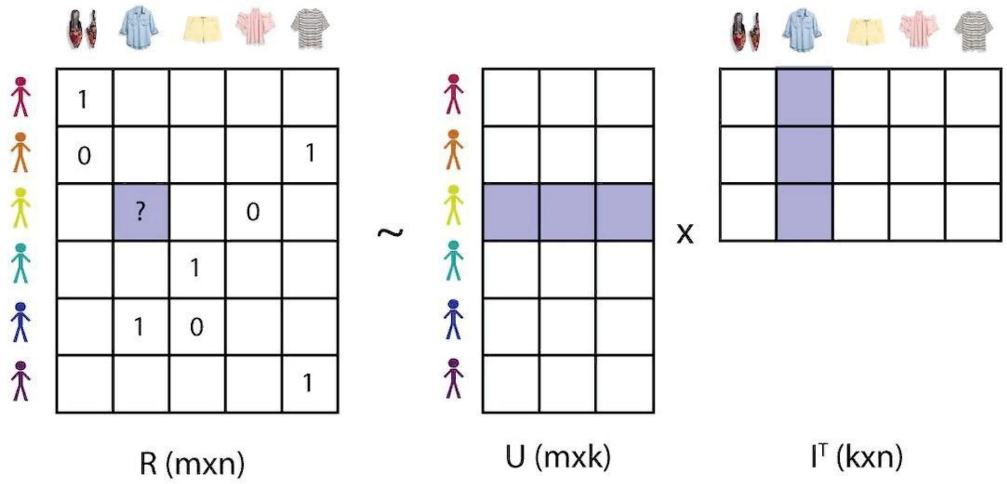
PyTorch Lightning

PyTorch

Matrix factorization

Assumption: observed interactions can be explained via:

- A small number of common patterns in human behavior
- Individual variations



Alternating Least Square

1) Add parameter pui - binarizing the values of rui

2) Add parameter cui - measure of confidence in pui

3) The optimization problem boils down to minimizing cost function

4) Time complexity is $O((M+N)K^3 + |\mathcal{R}|K^2)$

Element-Wise Alternating Least Squares model

1) To reduce the high time complexity with inverting a matrix, let us make a simplification (all zero entries have a same weight)

2) Also we can optimize parameters at the element level, taking it one step at a time

3) Time complexity will reduce to $O((M+N)K^2 + |R|K)$

Implicit Alternating Least Squares model

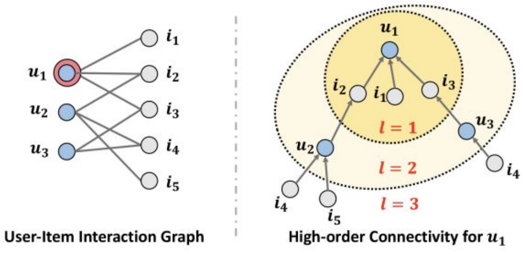
1) We fix user and item matrix solving individual linear regression problems in turn

2) Also we add 2 hyperparameters (unobserved weight instead measure of confidence and learning rate) to increase the convergence rate

3) Time complexity reduces to $O((M+N)K^2+|R|K)$

Neural Graph Collaborative Filtering

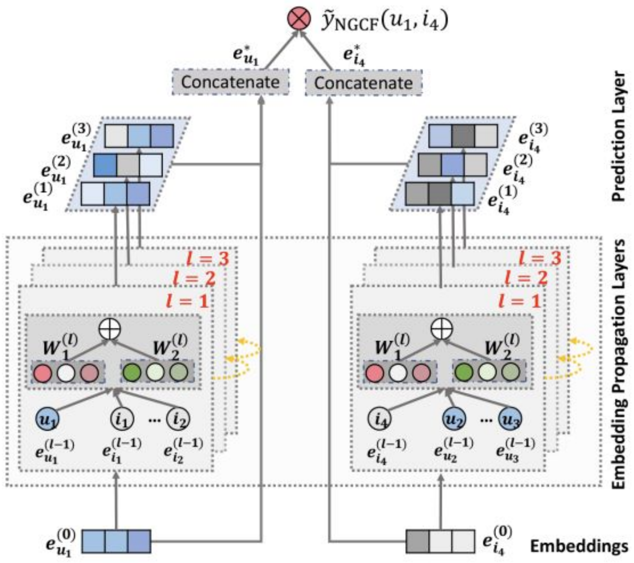
We can view interactions between items and users as Bipartite graph:



In math notations we can write this interactions as follows:

$$\mathbf{e}_u^{(l)} = \mathrm{LeakyReLU}\Big(\mathbf{m}_{u\leftarrow u}^{(l)} + \sum_{i\in\mathcal{N}_u} \mathbf{m}_{u\leftarrow i}^{(l)}\Big).$$

$$\begin{cases} \mathbf{m}_{u\leftarrow i}^{(l)} = p_{ui}\Big(\mathbf{W}_1^{(l)}\mathbf{e}_i^{(l-1)} + \mathbf{W}_2^{(l)}(\mathbf{e}_i^{(l-1)} \odot \mathbf{e}_u^{(l-1)})\Big), \\ \mathbf{m}_{u\leftarrow u}^{(l)} = \mathbf{W}_1^{(l)}\mathbf{e}_u^{(l-1)}, \end{cases}$$



The proposed architecture

Experimental methodology

Scenario:

Warm – start

Holdout construction:

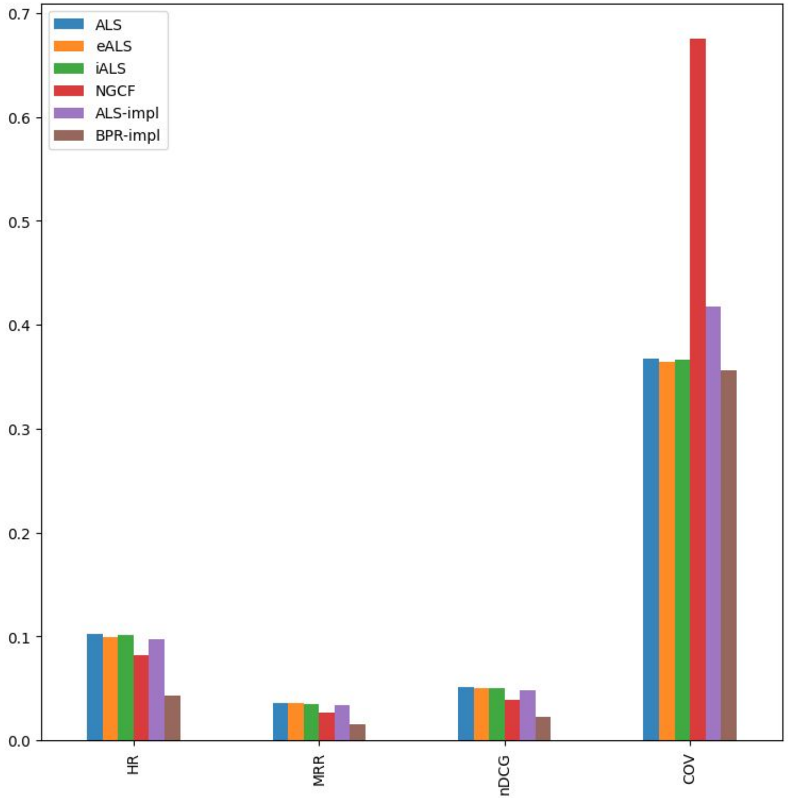
Time point split

Metrics:

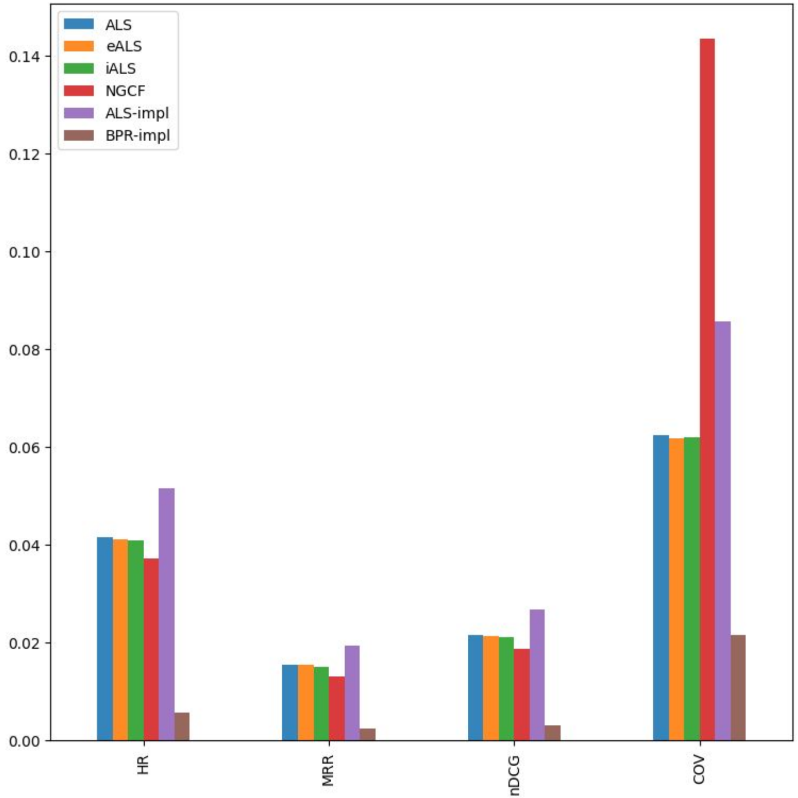
HR, MRR, nDCG, COV

Results: Metrics

MovieLens dataset

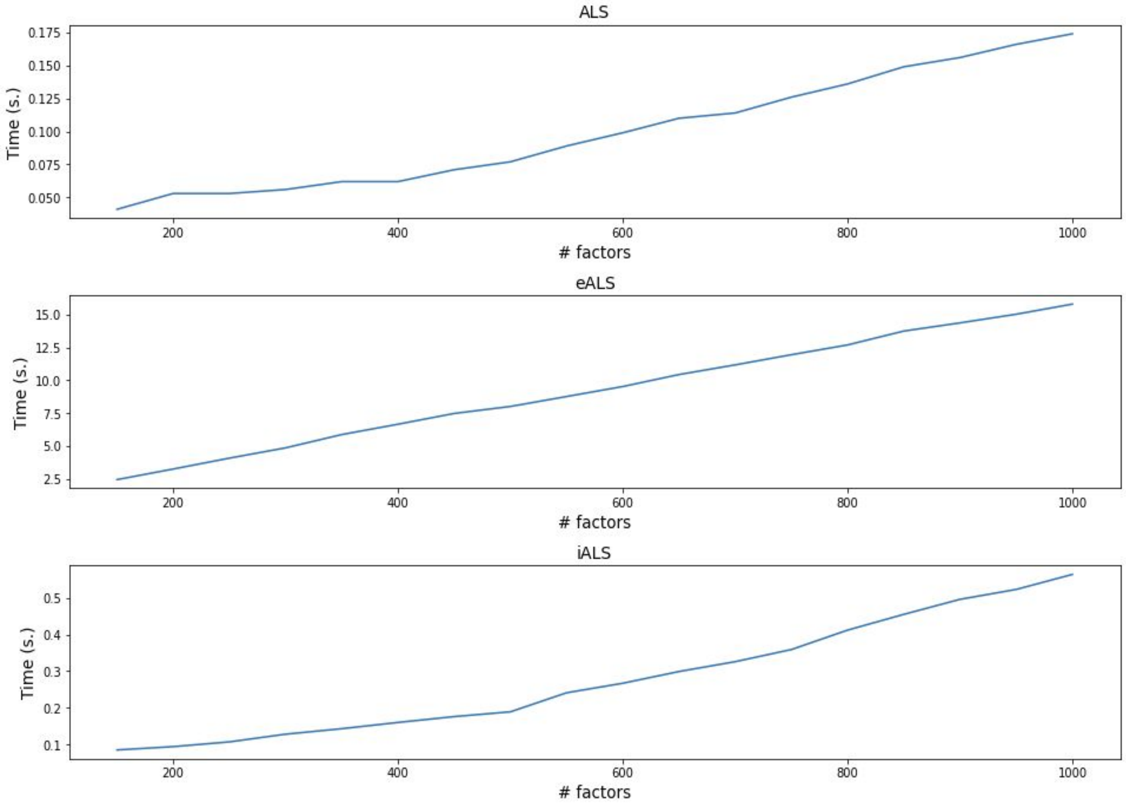


Yelp dataset

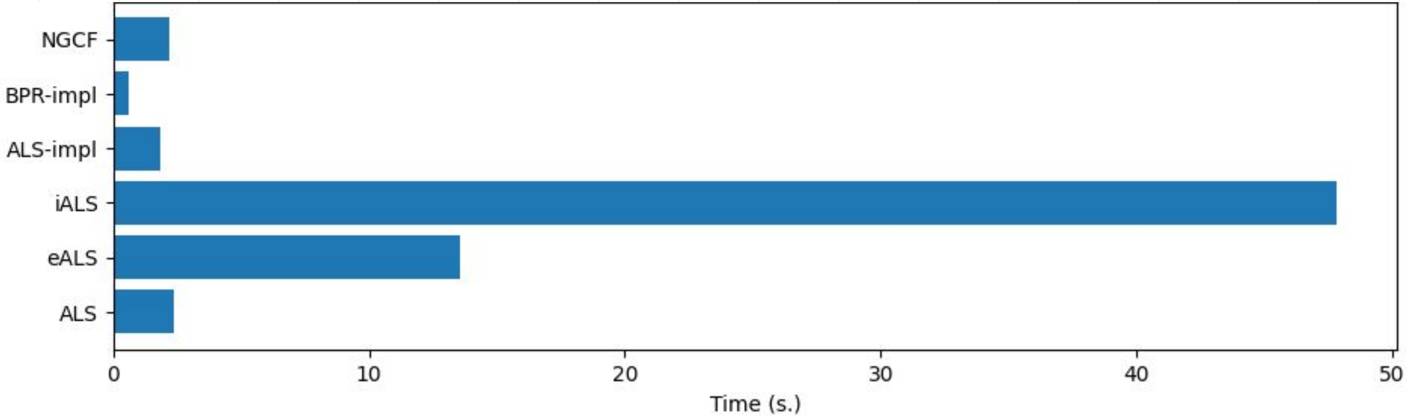


Time complexity: Theory vs Experiment

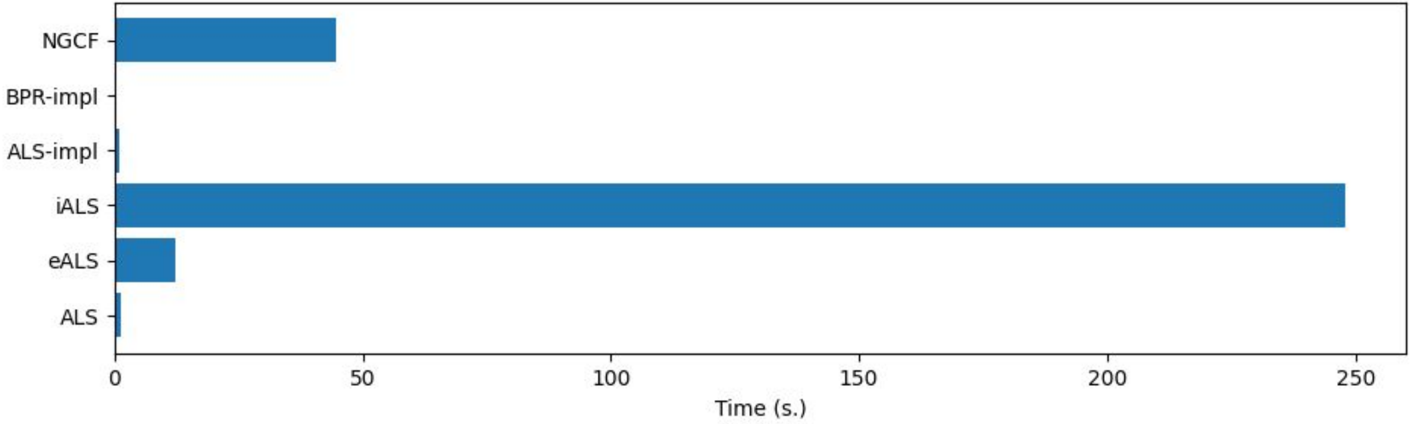
Model	Time complexity
ALS	$O((M+N)K^3 + \|\mathcal{R}\|K^2)$
eALS	$O((M+N)K^2 + \|R\|K)$
iALS	$O((M+N)K^2 + \|R\|K)$
NGCF	$O(\sum_{l=1}^{L} \|R^+\|d_l d_{l-1} + \sum_{l=1}^{L} \|R^+\|d_l)$



Computational speed



Movielens



Yelp

Conclusion

1) We have implemented a few different Collaborative Filtering models

2) We have compared computational time and the results on different metrics

3) Future work - implement ALS++ and one more neuro model

THANKS!

Nikolay Kotoyants

iALS
Presentation
Report

Konstantin Shlychkov

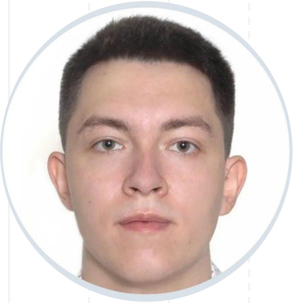
NGCF
Presentation
Project Organization

Alexander Kharitonov

ALS/eALS/NGCF
Presentation

Danil Gusak

Data prep/utils
Presentation
NGC



Gleb Mazanov

iALS
Presentation
Report

GitHub Repository

https://github.com/Mr6one/recsys-project-2023

Alternating Least Square

$$p_{ui} = \begin{cases} 1 & r_{ui} > 0 \\ 0 & r_{ui} = 0 \end{cases}$$

Binarizing the values

$$c_{ui} = 1 + \alpha r_{ui}$$

Measure of confidence in p_{ui}

$$\min_{x_\star, y_\star} \sum_{u,i} c_{ui}(p_{ui} - x_u^T y_i)^2 + \lambda \left(\sum_u \|x_u\|^2 + \sum_i \|y_i\|^2 \right)$$

Optimization problem - cost function

Alternating Least Square

$$x_u = (Y^T C^u Y + \lambda I)^{-1} Y^T C^u p(u) \qquad C^u \text{ where } C^u_{ii} = c_{ui}$$

Computing of user-factor

$$y_i = (X^T C^i X + \lambda I)^{-1} X^T C^i p(i) \qquad C^i \text{ where } C^i_{uu} = c_{ui}$$

Computing of user-factor is performed in time

Time complexity $O((M+N)K^3 + MNK^2)$

N is the overall number of non-zero observation

Element-Wise Alternating Least Squares model

To reduce the high time complexity with inverting a matrix, let us make a simplification (all zero entries in R have a same weight c_0)

$$Y^T C^u Y = c_0 Y^T Y + Y^T (C^u - C^0) Y$$

Time complexity of inversion reduces to $O(|\mathcal{R}|K)$

Element-Wise Alternating Least Squares model

Also we can optimize parameters at the element level

$$x_{uf} = \frac{\sum_{i=1}^{N} (riu - \hat{r}^{f}_{ui}) c_{ui} y_{if}}{\sum_{i=1}^{N} c_{ui} y_{if}^{2} + \lambda}$$

for user-factor

$$y_{if} = \frac{\sum_{i=1}^{N} (riu - \hat{r}^{f}_{ui}) c_{ui} x_{uf}}{\sum_{i=1}^{N} c_{ui} x_{uf}^{2} + \lambda}$$

for item-factor

Time complexity of computation is reducing to $O(MNK^2)$

Element-Wise Alternating Least Squares model

The full method time complexity reduces from

$$O((M+N)K^3 + MNK^2)$$ for ALS

to $$O((M+N)K^2 + |\mathcal{R}|K)$$ for eALS

Implicit Alternating Least Squares model

Idea – optimizing X while fixing Y and optimizing Y while fixing X.

$$x_{u^*} \leftarrow \Big(\sum_{(u^*,i,\beta,\alpha)} \alpha y_i \times y_i + \alpha_0 \sum_i y_i \times y_i + \lambda I \Big)^{-1} \sum_{(u^*,i,\beta,\alpha)} \alpha y_i \beta$$

$$y_{i^*} \leftarrow \Big(\sum_{(i^*,u,\beta,\alpha)} \alpha x_u \times x_u + \alpha_0 \sum_u x_u \times x_u + \lambda I \Big)^{-1} \sum_{(i^*,u,\beta,\alpha)} \alpha x_u \beta$$

Alpha and beta are hyperparameters and usually equal to 1

$$\mathcal{O}(d|S| + d^2 (|U| + |I|))$$

Implicit Alternating Least Squares model

Computation time complexity is

$$\mathcal{O}\left(d|S| + d^2\left(|U| + |I|\right)\right)$$

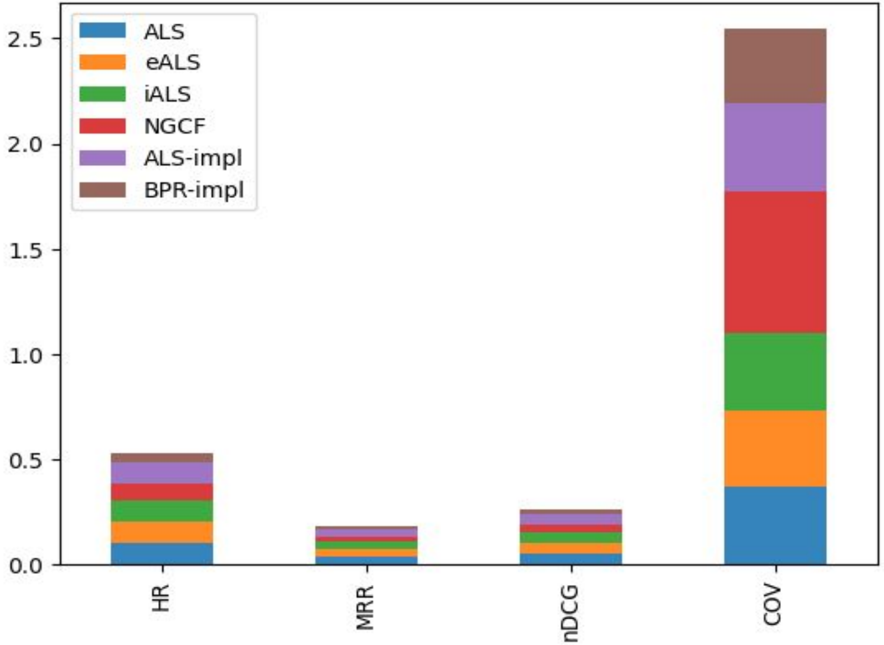
where

S - a set of positive user-item pairs

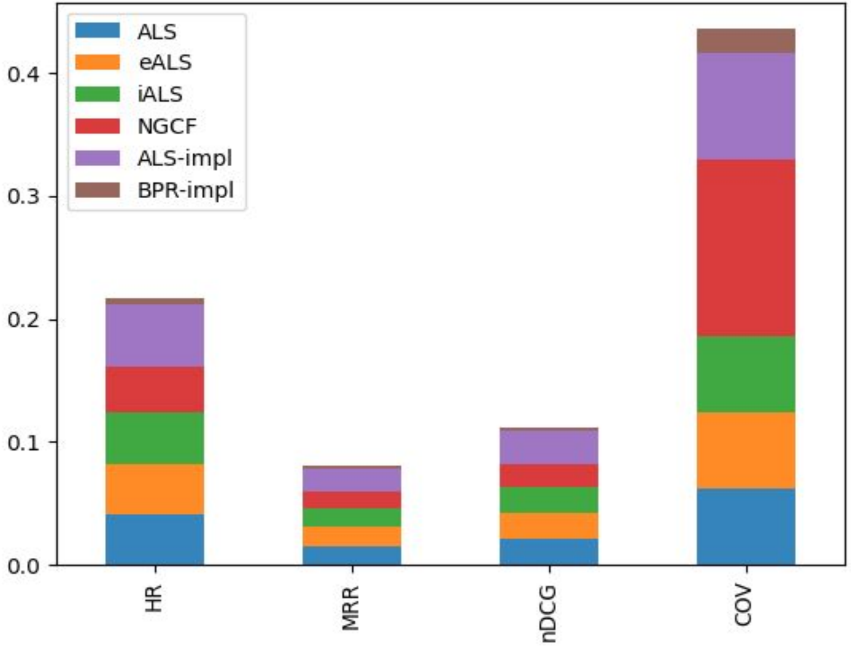
d - dimension of embeddings

Results: Metrics

MovieLens dataset



Yelp dataset



Results: Yelp dataset