Machine Learning with Graphs (MLG)

HW3: Model Design & Comparison for Recommendation Models

Deadline: **2020.05.31 (Sat.)** 23:59

Submission: Code (.py/.ipynb) and Report (PDF)

標題	觀看次數	觀看小時	
MLG-2020 01-5 Centrality Prediction	127	28.5079	HW1
MLG-2020 00 Course Intro	97	8.7164	
MLG-2020 01 Graph Structure	86	15.6874	HW1
MLG-2020 01-3 Centrality Analysis	83	15.4437	HW1
MLG-2020 02-1 Network Properties	61	16.1875	HW1
MLG-2020 03-1 Link Prediction	60	18.5091	HW2
MLG-2020 HW2	60	4.3949	HW2
MLG-2020 01-4 Eigenvector Centrality	59	7.7755	HW1
MLG-2020 04-1 High-Order Link Prediction	56	8.4266	HW2
MLG-2020 01-2 Graph Search	55	6.9471	HW1
MLG-2020 03-2 LP on Attributed Graphs	48	10.7277	HW2
MLG-2020 02-2 Graph Generation: ER Model	47	10.8395	noHW
MLG-2020 02-3 Graph Generation: BA Model	46	5.2461	noHW
MLG-2020 02-4 Graph Generation: WS Model	40	4.3228	noHW
MLG-2020 05-1 Community Detection: Basic	38	6.9724	noHW
MLG-2020 04-2 Signed Link Prediction	36	2.7258	noHW
MLG-2020 06-1 RecSys: Collaborative Filtering	25	4.7612	
MLG-2020 05-3 Community Detection: Louvain & LPA	21	3.3071	noHW
MLG-2020 05-2 Community Detection: Edge-Removal	21	6.0457	noHW
MLG-2020 04-3 Dynamic Link Prediction	17	1.478	noHW
MLG-2020 04-4 Link Prediction on Knowledge Graphs	15	4.2244	noHW
MLG-2020 06-2 RecSys: Matrix Factorization	14	3.364	
MLG-2020 07-1 RecSys: BPR Bayesian Personalized Ranking	9	2.3756	
MLG-2020 06-3 RecSys: Factorization Machine	7	1.5435	

The Main Expected Performance Table

	LON-A			LYC-R		
	LogLoss	**AUC**	**NDCG@5**	**LogLoss**	**AUC**	**NDCG@5**
UCF-s						
UCF-p						
ICF-s						
ICF-p						
MF						
FM						
BPR-MF						
BPR-FM						
GBDT+LR						
XGB+LR						
FNN						
IPNN						
OPNN						
PIN						
CCPM						
NeuMF						
WD						
DeepCross						
NFM						
DeepFM						
5選3 (自行加rows)						
Your own NN model						

10 Typical RecSys Methods

10 NN-based RecSys Methods

3 Recent NN-based Methods

你自己設計的NN方法

RecSys Models to be Compared

10 Typical Approaches

1) User-based CF [**UCF-s**] (cosine as similarity)
2) User-based CF [**UCF-p**] (Pearson correlation as similarity)
3) Item-based CF [**ICF-s**] (cosine as similarity)
4) Item-based CF [**ICF-p**] (Pearson correlation as similarity)
5) Matrix Factorization [**MF**] (varying k-dim latent factor)
6) Factorization Machine [**FM**] (varying k-dim latent factor)
7) Matrix Factorization with BPR [**BPR-MF**]
8) Factorization Machine with BPR [**BPR-FM**]
9) Pre-training via GBDT for LR [**GBDT-LR**]
10) Pre-training via XGBoost for LR [**XGB-LR**]

Lecture 6

Lecture 7

RecSys Models to be Compared

- **10** NN-based approaches

1) FM-supported Neural Networks [**FNN**]
2) Product-based Neural Networks
   - Inner-Product NN [**IPNN**]
   - Outer-Product NN [**OPNN**]
   - Product-network in Network [**PIN**]
3) Convolutional Click Prediction Model [**CCPM**]
4) NCF Neural Matrix Factorization [**NeuMF**]
5) Wide&Deep [**WD**]
6) Deep Crossing [**DeepCross**]
7) Neural Factorization Machine [**NFM**]
8) Deep Factorization Machine [**DeepFM**]

Lecture 8

RecSys Models to be Compared

Select 3 from 5 Recent NN-based approaches:

1) Attentional Factorization Machines (**AFM**)
   Learning the Weight of Feature Interactions via Attention Networks (IJCAI 2017)
2) Collaborative Memory Networks (**CMN**)
   Collaborative Memory Network for Recommendation Systems (SIGIR 2018)
3) **xDeepFM**
   Combining Explicit and Implicit Feature Interactions for RecSys (KDD 2018)
4) Deep Interest Network (**DIN**)
   Deep Interest Network for Click-Through Rate Prediction (AAAI 2019)
5) **DeepGBM**
   A Deep Learning Framework Distilled by GBDT for Online Prediction Tasks (KDD 2019)

Evaluation Metrics & Datasets

- TripAdvisor Datasets:
  - **LON-A**: tourist attractions in London
  - **NYC-R**: restaurants in New York City
- Evaluation metrics:
  (1) **avg** of **AUC**, (2) **avg** of **LogLoss**, (3) **avg** of **NDCG@5**

Dataset	User#	User Feature#	Item#	Item Feature#	Interaction#
LON-A	16, 315	3, 230	953	4, 731	136, 978
NYC-R	15, 232	3, 230	6, 258	10, 411	129, 964

Side Information	Features (Category#)
LON-A/NYC-R User Feature	Age (6), Gender (2), Expert Level (6), Traveler Styles (18), Country (126), City (3, 072)
LON-A Attraction Feature	Attributes (89), Tags (4, 635), Rating (7)
NYC-R Restaurant Feature	Attributes (100), Tags (10, 301), Price (3), Rating (7)

Data Columns

- Column = Unnamed:0 = row ID
- Rating
  - rtime: rating time
  - rquote: comments along with rating
  - rrate: rating score
  - rid: rating ID

Item
- iid: item ID
- iattribute: item attributes
- iprice: item price (for only NYC-R)
- irating: item avg rating
- itag: item's tags

User
- uage: user age
- ugender: user gender
- ucity: user's located city
- ucountry: user's located country
- uid_index: user ID
- ulevel: user's expert level at TripAdvisor
- ustype: user's traveler style

Please ignore all of the remaining columns that do not list here

Evaluation Settings

- **Data preprocessing**
  - Retain users/items with at least five ratings only
- **Data splitting**
  - Test data: **the latest 20%** interactions (by time) of each user
  - Randomly split the remaining data into training (70%) and validation (10%) sets
  - Validation set is for hyperparameter tuning
- Transform the ratings into **binary implicit feedback as ground truth**, indicating whether the user has interacted with the specific item
- **[Important!!]** Be sure to **fairly** do all comparisons under the same experimental settings

Task Requirements

- Q1: Compared with the typical methods, can our NN-based approaches achieve comparable accuracy? Why?
  - Are recent NN-based methods even better? Why?
- Q2: Are there any hyperparameters in each model that significantly affect the performance?
  - You need to conduct hyperparameter studies for some modes, find the best hyperparamters, and explain why such settings are good
- Q3: Can you create a new end-to-end NN that combine the advantages of nicely-performed methods to beat all methods?
  - You **MUST** at least devise one novel NN-based method by yourself, and have it compared with all methods
  - No matter you beat them or fail, explain the possible reasons
- Q4: What if I cannot successfully complete **some** (e.g., 8, 7, 2) of 10 typical, 10 NN-based, 3/5 recent method?
  - Just try your best to have 13 compared method. Do as many as you can.
  - You own method is definitely required

Reference Packages (but not limited)

- Surprise https://github.com/NicolasHug/Surprise
- Spotlight https://github.com/maciejkula/spotlight
- LightFM https://github.com/lyst/lightfm/
- DeepCTR https://github.com/shenweichen/DeepCTR
- NeuRec https://github.com/wubinzzu/NeuRec
- RecQ https://github.com/Coder-Yu/RecQ
- **Bonus:** implement all required models by yourself using PyTorch

- We recommend you to **read the original papers of all required models** so that you can understand to come up with your own method, and make systematic and correct comparison
  - At least you can find the key hyperparameters in the papers

HW3 Submission

- HW3 Report + Code submission via **Moodle**
  - Deadline: **May 31, (Sat) 2020, 23:59**
  - Submit your code: **.py** or **.ipynb** (preferred)
  - Submit report (PDF): $\geq$**15** pages (you cannot include code in report)
- Content in the report
  - **1) Introduction**
  - **2) Methodology**: briefly describe all of the compared methods, and **describe the details of your own method**
  - **3) Experimental analysis**, along with analysis and insights
    - Report your experimental settings, hyperparameter setting of each method
    - Compare and report the required methods and your own method
    - Explain WHY your prediction is so GOOD or so BAD!
    - Present any insights based on your results
    - Do hyperparameter analysis
    - Refer to the slide “Task Requirements”
  - **4) Conclusions**
    - Explain the **novelty** of your method, summarize your findings
    - Point out how to improve in the future
  - **5) Citations** (if you use any methods or papers)