Types: Collaborative filtering-based, Content-based, Knowledge-based, Graph-based, Hybrid

Challenges (with importance level):

- (High) **Cold-start issue:** handle new users or items that have no rating data.
  - GHRS model
- (Normal) **Online training:** update the model based on new ratings.
  - paper #ToRead
- (High) **Trend items:** consider an advantage for new and trending items.
- (Normal) **Overspecialization:** suggest diverse and surprising items.
- (Low) **Predictability:** suggest fresh items that user haven't seen.
- (Low) **Real-time:** handle real-time changes in user's preference (mostly for suggesting similar items based on user info).
  - medium article #ToRead
- (Low) **Interactive learning:** update the model based on user's interest in the model's suggestions.
  - bandits article

Similar Items Recommendation

weighted feature cosine similarity, can train weights offline or online (bandits article)

w2v + Approximate nearest neighbor (ScANN)

Design 1

- **Item Feature Generator:** Image and Text features, User-Item matrix features, etc.
  - Can extract user-item matrix features (e.g., using matrix factorization, SVD) periodically (e.g., once a day), and only use side information features for new items.
- **Similar Items:** dot product, Approximate nearest neighbor
- **User Log:** viewed, clicked, liked, saved, bought items
- **Recommended Items:** find related items based on user log

$$predicted\ rating(i) = \sum_j w_j \times sim(i,j),$$

$$where\ w_j = \begin{cases} 3, if\ item\ j\ is\ related\ to\ browsing \\ 4, if\ item\ j\ is\ related\ to\ searching \\ rates, if\ item\ j\ is\ related\ to\ rating \end{cases}$$

$$sim(i,j) = \frac{Item(i) \cdot Item(j)}{|Item(i)| * |Item(j)|},$$

$$where\ the\ item\ vector\ representation\ is\ computed\ from\ ALS\ recommender$$

Design 2 (TencentRec)

- **Item Feature Generator:** Image and Text features, User-Item matrix features
    - User-Item matrix is directly used as feature vector to calculate similarities.
- **Similar Items:**

$$\text{co-rating}(i_p, i_q) = \min(r_{u,p}, r_{u,q})$$

$$sim(i_p, i_q) = \frac{\text{pairCount}(i_p, i_q)}{\sqrt{\text{itemCount}(i_p)}\sqrt{\text{itemCount}(i_q)}}$$

where

$$\text{itemCount}(i_p) = \sum r_{u,p}$$

$$\text{pairCount}(i_p, i_q) = \sum_{u \in U} \text{co-rating}(i_p, i_q)$$

- **User Log:** viewed, clicked, liked, saved, bought items
- **Recommended Items:** find related items based on user log

$$\hat{r}_{u,p} = \frac{\sum_{i_q \in N^k(i_p)} sim(i_p, i_q) r_{u,i_q}}{\sum_{i_q \in N^k(i_p)} sim(i_p, i_q)}$$

Youtube Recommender with Deep Learning model

https://dl.acm.org/doi/abs/10.1145/3298689.3346997 (2019)

https://dl.acm.org/doi/10.1145/2959100.2959190 (2016)

Real-time recommenders

- Eugeneyan article
- Java + etc, API seems ok (2020)
- Scala + Kafka, good and simple API (2016)
- Python, API commands not clear yet (2018)
- Python, super complete and complex, uses AWS (2022)
- Medium article

Simple starting library with baseline methods (https://github.com/NicolasHug/Surprise)

Truncated SVD (https://analyticsindiamag.com/singular-value-decomposition-svd-application-recommender-system/) (https://towardsdatascience.com/recommender-system-singular-value-decomposition-svd-truncated-svd-97096338f361)

List of RS (open-source, research, benchmarks) (https://github.com/grahamjenson/list_of_recommender_systems)

PredictionIO (https://github.com/apache/predictionio) (Abandoned)

- Similar products service (based on item categories and user views)
- E-commerce RS

Papers (https://github.com/hongleizhang/RSPapers)

Microsoft repository (https://github.com/microsoft/recommenders)

Algo	MAP	nDCG@k	Precision@k	Recall@k	RMSE	MAE	R^2	Explained Variance
ALS	0.004732	0.044239	0.048462	0.017796	0.965038	0.753001	0.255647	0.251648
BiVAE	0.146126	0.475077	0.411771	0.219145	N/A	N/A	N/A	N/A
BPR	0.132478	0.441997	0.388229	0.212522	N/A	N/A	N/A	N/A
FastAI	0.025503	0.147866	0.130329	0.053824	0.943084	0.744337	0.285308	0.287671
LightGCN	0.088526	0.419846	0.379626	0.144336	N/A	N/A	N/A	N/A
NCF	0.107720	0.396118	0.347296	0.180775	N/A	N/A	N/A	N/A
SAR	0.110591	0.382461	0.330753	0.176385	1.253805	1.048484	-0.569363	0.030474
SVD	0.012873	0.095930	0.091198	0.032783	0.938681	0.742690	0.291967	0.291971

BiVAE:

- https://github.com/PreferredAI/bi-vae
- https://github.com/microsoft/recommenders/blob/main/examples/02_model_collaborative_filtering/cornac_bivae_deep_dive.ipynb

MovieLens 1M benchmark (https://paperswithcode.com/sota/collaborative-filtering-on-movielens-1m)

GLocal-K:

- https://github.com/usydnlp/Glocal_K

Graph-based hybrid RS:

- https://github.com/hadoov/GHRS

IMC-GAE (graph autoencoder):

- https://github.com/swtheing/imc-gae

Amazon product data (score, review, product metadata) (https://paperswithcode.com/dataset/amazon-product-data)