Recommender Systems in Large Scale ML

Шугаепов Ильнур

VK.com
Performance Advertising Team
ilnur.shug@gmail.com

Higher School of Economics, 2020

RS [16]

Recommender Systems (RSs) are software tools and techniques that provide suggestions for items that are most likely of interest to a particular user

RS [16]

Recommender Systems (RSs) are software tools and techniques that provide suggestions for items that are most likely of interest to a particular user

Notations

- U — set of subjects (users)
- I — set of objects (items)
- $D = \{(u_t, i_t, y_t)\}_{t=1}^{m} \subset U \times I \times Y$ — transactions, where Y — set of transactions descriptions

Applications

- E-commerce
  - U - clients of online shop
  - I - products (books, movies, music, etc)
  - $r_{ui} = \mathbb{1}\{u \text{ bought } i\}$

Applications

- E-commerce
  - U - clients of online shop
  - I - products (books, movies, music, etc)
  - $r_{ui} = \mathbb{1}\{u \text{ bought } i\}$
- Social network
  - U - users of the site
  - I - posts, communities
  - $r_{ui} = \mathbb{1}\{u \text{ visited } i\}$

Applications

- E-commerce
  - U - clients of online shop
  - I - products (books, movies, music, etc)
  - $r_{ui} = \mathbb{1}\{u \text{ bought } i\}$
- Social network
  - U - users of the site
  - I - posts, communities
  - $r_{ui} = \mathbb{1}\{u \text{ visited } i\}$
- Movies recommendation
  - U - clients of the platform
  - I - movies
  - $r_{ui} =$ rating u gave to i

Implicit vs. Explicit Feedback

Users transactions (feedback) D can be devided into two types

	Explicit	**Implicit**
r_{ui}	explicit rating for item i by user u	fact (number of times) that u interacted with i

Implicit vs. Explicit Feedback

Users transactions (feedback) D can be devided into two types

	Explicit	**Implicit**
r_{ui}	explicit rating for item i by user u	fact (number of times) that u interacted with i
Gathering complexity		

Implicit vs. Explicit Feedback

Users transactions (feedback) D can be devided into two types

	Explicit	**Implicit**
r_{ui}	explicit rating for item i by user u	fact (number of times) that u interacted with i
Gathering complexity	Hard to collect	Easy to collect

Implicit vs. Explicit Feedback

Users transactions (feedback) D can be devided into two types

	Explicit	**Implicit**
r_{ui}	explicit rating for item i by user u	fact (number of times) that u interacted with i
Gathering complexity	Hard to collect	Easy to collect
Amount		

Implicit vs. Explicit Feedback

Users transactions (feedback) D can be devided into two types

	Explicit	**Implicit**
r_{ui}	explicit rating for item i by user u	fact (number of times) that u interacted with i
Gathering complexity	Hard to collect	Easy to collect
Amount	Small	Large

Implicit vs. Explicit Feedback

Users transactions (feedback) D can be devided into two types

	Explicit	**Implicit**
r_{ui}	explicit rating for item i by user u	fact (number of times) that u interacted with i
Gathering complexity	Hard to collect	Easy to collect
Amount	Small	Large

Remark

It is straighforward to conver explicit to implicit, for example:

- *if r_{ui} is a movie rating then $p_{ui} = \mathbb{1}\{r_{ui} \geq 3\}$ - implicit feedback*

If not stated otherwise we assume explicit feedback

Table of Contents

1. Evaluation

2. Matrix Completion

3. Link Prediction

4. Session-based Recommendations

5. Learning to Rank

6. Resume

7. RecSys at VK

Requirements

- System should be able to compute $\rho(u, i), \rho(u, u'), \rho(i, i')$, where ρ — similarity (relevance) function
- Given user u system should be able to rank I according to $\rho(u, \cdot)$

- We have different formulations of RS problem thus different solutions
- Main goal is to compare different solutions

Protocol

1. Order all transactions D by time
2. Split D into D_{train} / D_{valid} / D_{test} sets by timestamp
3. Fit models on D_{train}
4. HPO on D_{valid}
5. Report resulting metrics on D_{test}

Protocol

1. Order all transactions D by time
2. Split D into D_{train} / D_{valid} / D_{test} sets by timestamp
3. Fit models on D_{train}
4. HPO on D_{valid}
5. Report resulting metrics on D_{test}

Problems

- Cold-start users — users which appear only in D_{test} set
- Cold-start items — items which appear only in D_{test} set

Protocol

1. Order all transactions D by time
2. Split D into D_{train} / D_{valid} / D_{test} sets by timestamp
3. Fit models on D_{train}
4. HPO on D_{valid}
5. Report resulting metrics on D_{test}

Problems

- Cold-start users — users which appear only in D_{test} set
- Cold-start items — items which appear only in D_{test} set

Common solutions

- Leave only users and items which appear in all sets

Protocol

1. Split D into set of sessions $\mathcal{S} = \{S_u\}_{u \in U}$
2. For each user u split S_u into $S_{u,train}, S_{u,valid}, S_{u,test}$ (using timestamp)
3. Therefore $D_s = \cup_{u \in U} S_{u,s}$ for $s \in \{train, valid, test\}$
4. Fit models on D_{train}
5. HPO on D_{valid}
6. Report resulting metrics on D_{test}

Protocol

1. Split D into set of sessions $\mathcal{S} = \{S_u\}_{u \in U}$
2. For each user u split S_u into $S_{u,train}, S_{u,valid}, S_{u,test}$ (using timestamp)
3. Therefore $D_s = \cup_{u \in U} S_{u,s}$ for $s \in \{train, valid, test\}$
4. Fit models on D_{train}
5. HPO on D_{valid}
6. Report resulting metrics on D_{test}

Problems

- Peeking into the future

Protocol

1. Split D into set of sessions $\mathcal{S} = \{S_u\}_{u \in U}$
2. For each user u split S_u into $S_{u,train}, S_{u,valid}, S_{u,test}$ (using timestamp)
3. Therefore $D_s = \cup_{u \in U} S_{u,s}$ for $s \in \{train, valid, test\}$
4. Fit models on D_{train}
5. HPO on D_{valid}
6. Report resulting metrics on D_{test}

Problems

- Peeking into the future

Common solutions

- Use previous scenario instead

Metrics

- For each user u we have a set of N_u ground truth relevant items $D_u = \{i_1, i_2, \dots, i_{N_u}\}$ and
- List of Q_u recommended items (according to $\rho(u, \cdot)$) $R_u = \{r_1, r_2, \dots, r_{Q_u}\}$, in order of decreasing relevance

[1] https://en.wikipedia.org/wiki/Evaluation_measures_(information_retrieval)

- For each user u we have a set of N_u ground truth relevant items $D_u = \{i_1, i_2, \dots, i_{N_u}\}$ and
- List of Q_u recommended items (according to $\rho(u, \cdot)$) $R_u = \{r_1, r_2, \dots, r_{Q_u}\}$, in order of decreasing relevance

Popular Ranking Metrics[1]

Metric	**Definition**	**Notes**
Precision@k	$\frac{1}{\lvert U \rvert} \sum_{u \in U} \frac{1}{k} \sum_{i=1}^{\min(k, Q_u)} \mathbb{1}\{r_i \in D_u\}$	(# of recommended items @k that are relevant) / (# of recommended items @k)

[1] https://en.wikipedia.org/wiki/Evaluation_measures_(information_retrieval)

Metrics

- For each user u we have a set of N_u ground truth relevant items $D_u = \{i_1, i_2, \ldots, i_{N_u}\}$ and
- List of Q_u recommended items (according to $\rho(u, \cdot)$) $R_u = \{r_1, r_2, \ldots, r_{Q_u}\}$, in order of decreasing relevance

Popular Ranking Metrics[1]

Metric	**Definition**	**Notes**
Precision@k	$\frac{1}{\lvert U\rvert} \sum_{u \in U} \frac{1}{k} \sum_{i=1}^{\min(k, Q_u)} \mathbb{1}\{r_i \in D_u\}$	(# of recommended items @k that are relevant) / (# of recommended items @k)
Recall@k	$\frac{1}{\lvert U\rvert} \sum_{u \in U} \frac{1}{\lvert D_u\rvert} \sum_{i=1}^{\min(k, Q_u)} \mathbb{1}\{r_i \in D_u\}$	(# of recommended items @k that are relevant) / (total # of relevant items)

[1] https://en.wikipedia.org/wiki/Evaluation_measures_(information_retrieval)

Metrics

- For each user u we have a set of N_u ground truth relevant items $D_u = \{i_1, i_2, \dots, i_{N_u}\}$ and
- List of Q_u recommended items (according to $\rho(u, \cdot)$) $R_u = \{r_1, r_2, \dots, r_{Q_u}\}$, in order of decreasing relevance

Popular Ranking Metrics[1]

Metric	**Definition**	**Notes**
Precision@k	$\frac{1}{\lvert U\rvert} \sum_{u \in U} \frac{1}{k} \sum_{i=1}^{\min(k, Q_u)} \mathbb{1}\{r_i \in D_u\}$	(# of recommended items @k that are relevant) / (# of recommended items @k)
Recall@k	$\frac{1}{\lvert U\rvert} \sum_{u \in U} \frac{1}{\lvert D_u\rvert} \sum_{i=1}^{\min(k, Q_u)} \mathbb{1}\{r_i \in D_u\}$	(# of recommended items @k that are relevant) / (total # of relevant items)
HR@k	$\frac{1}{\lvert U\rvert} \sum_{u \in U} \mathbb{1}\{R_{u,1:k} \cap D_u \neq \emptyset\}$	(# of times top k contains relevant) / (# of users)

[1] https://en.wikipedia.org/wiki/Evaluation_measures_(information_retrieval)

- For each user u we have a set of N_u ground truth relevant items $D_u = \{i_1, i_2, \dots, i_{N_u}\}$ and
- List of Q_u recommended items (according to $\rho(u, \cdot)$) $R_u = \{r_1, r_2, \dots, r_{Q_u}\}$, in order of decreasing relevance

Popular Ranking Metrics[1]

Metric	Definition	Notes
Precision@k	$\frac{1}{\lvert U\rvert} \sum_{u \in U} \frac{1}{k} \sum_{i=1}^{\min(k, Q_u)} \mathbb{1}\{r_i \in D_u\}$	(# of recommended items @k that are relevant) / (# of recommended items @k)
Recall@k	$\frac{1}{\lvert U\rvert} \sum_{u \in U} \frac{1}{\lvert D_u\rvert} \sum_{i=1}^{\min(k, Q_u)} \mathbb{1}\{r_i \in D_u\}$	(# of recommended items @k that are relevant) / (total # of relevant items)
HR@k	$\frac{1}{\lvert U\rvert} \sum_{u \in U} \mathbb{1}\{R_{u,1:k} \cap D_u \neq \emptyset\}$	(# of times top k contains relevant) / (# of users)
MAP	$\frac{1}{\lvert U\rvert} \sum_{u \in U} \frac{1}{N_u} \sum_{i=1}^{Q_u} \frac{1}{i} \mathbb{1}\{r_i \in D_u\}$	how many of the recommended documents are in the set of true relevant documents

[1] https://en.wikipedia.org/wiki/Evaluation_measures_(information_retrieval)

Metrics

- For each user u we have a set of N_u ground truth relevant items $D_u = \{i_1, i_2, \ldots, i_{N_u}\}$ and
- List of Q_u recommended items (according to $\rho(u, \cdot)$) $R_u = \{r_1, r_2, \ldots, r_{Q_u}\}$, in order of decreasing relevance

Popular Ranking Metrics[1]

Metric	**Definition**	**Notes**
Precision@k	$\frac{1}{\lvert U\rvert} \sum_{u \in U} \frac{1}{k} \sum_{i=1}^{\min(k, Q_u)} \mathbb{1}\{r_i \in D_u\}$	(# of recommended items @k that are relevant) / (# of recommended items @k)
Recall@k	$\frac{1}{\lvert U\rvert} \sum_{u \in U} \frac{1}{\lvert D_u\rvert} \sum_{i=1}^{\min(k, Q_u)} \mathbb{1}\{r_i \in D_u\}$	(# of recommended items @k that are relevant) / (total # of relevant items)
HR@k	$\frac{1}{\lvert U\rvert} \sum_{u \in U} \mathbb{1}\{R_{u,1:k} \cap D_u \neq \emptyset\}$	(# of times top k contains relevant) / (# of users)
MAP	$\frac{1}{\lvert U\rvert} \sum_{u \in U} \frac{1}{N_u} \sum_{i=1}^{Q_u} \frac{1}{i} \mathbb{1}\{r_i \in D_u\}$	how many of the recommended documents are in the set of true relevant documents
NDCG		

[1] https://en.wikipedia.org/wiki/Evaluation_measures_(information_retrieval)

- Precision@k, Recall@k, HR@k — the order of the recommendations is not taken into account
- MAP, NDCG — metric takes into account the order of the recommendations

- Precision@k, Recall@k, HR@k — the order of the recommendations is not taken into account
- MAP, NDCG — metric takes into account the order of the recommendations

Q: Why not to use ROC AUC?

Table of Contents

1. Evaluation

2. Matrix Completion
   - Formulation
   - Singular Value Decomposition
   - Alternating Least Squares
   - Large Scale Matrix Factorization

3. Link Prediction

4. Session-based Recommendations

5. Learning to Rank

6. Resume

7. RecSys at VK

Matrix Completion

Aggregated data:

- $R = (r_{ui})_{u \in U, i \in I}$ — cross-tabulation matrix, where
- $r_{ui} = agg\{(u_t, i_t, y_t) \in D \mid u_t = u \wedge i_t = i\}$

Matrix Completion

Aggregated data:

- $R = (r_{ui})_{u \in U, i \in I}$ — cross-tabulation matrix, where
- $r_{ui} = agg\{(u_t, i_t, y_t) \in D \mid u_t = u \wedge i_t = i\}$

Task:

- Fill missing values r_{ui}

Matrix Completion

Aggregated data:

- $R = (r_{ui})_{u \in U, i \in I}$ — cross-tabulation matrix, where
- $r_{ui} = agg\{(u_t, i_t, y_t) \in D \mid u_t = u \wedge i_t = i\}$

Task:

- Fill missing values r_{ui}

Examples of r_{ui}:

- rating from user u to movie i
- number of times user u visited page i

- Content-based methods
- Collaborative filtering
  - User/Item based
  - Matrix Factorizations (SVD, PMF [12], ALS [8, 9])
- Neural Architectures (NCF [7], CB2CF [1])

Definition (Latent Factor Model via Matrix Factorization)

Given data D, our goal is to find matrices $P = (p_{ut})_{|U| \times |T|}$ and $Q = (q_{it})_{|I| \times |T|}$, where T - set of latent factors ($|T| \ll |U|, |T| \ll |I|$) such that

$$R = P \Delta Q^T,$$

$\Delta = diag(\pi_1, \dots, \pi_{|T|})$

Low-rank approximation

$$R_k \equiv P_k \Sigma_k Q_k^T \quad : \quad \|R - R_k\|_F \to \min,$$

where k - rank

Low-rank approximation

$$R_k \equiv P_k \Sigma_k Q_k^T \quad : \quad \|R - R_k\|_F \to \min,$$

where k - rank

Remark

R_k is the best rank-k approximation of the matrix R in terms of F norm (RMSE)[a]

[a] https://en.wikipedia.org/wiki/Low-rank_approximation

Low-rank approximation

$$R_k \equiv P_k \Sigma_k Q_k^T \quad : \quad \|R - R_k\|_F \to \min,$$

where k - rank

Remark

R_k is the best rank-k approximation of the matrix R in terms of F norm (RMSE)[a]

[a] https://en.wikipedia.org/wiki/Low-rank_approximation

Q: Before we apply SVD we need to fill unobserved values in R.
How to do that?

Low-rank approximation

$$R_k \equiv P_k \Sigma_k Q_k^T \quad : \quad \|R - R_k\|_F \to \min,$$

where k - rank

Remark

R_k is the best rank-k approximation of the matrix R in terms of F norm (RMSE)[a]

[a] `https://en.wikipedia.org/wiki/Low-rank_approximation`

Q: Before we apply SVD we need to fill unobserved values in R. How to do that?

A: Popular choices

- $r_{ui} = 0$ if value is unobserved

- Missing values in R are mixture of true negative values and abscence of interactions. Therefore $r_{ui} = 0$ is not best possible choice.

- Missing values in R are mixture of true negative values and absence of interactions. Therefore $r_{ui} = 0$ is not best possible choice.
- **Q:** Suppouse that $R_k = R$. Is this model useful for recommendations?

- Missing values in R are mixture of true negative values and abscence of interactions. Therefore $r_{ui} = 0$ is not best possible choice.
- **Q:** Suppouse that $R_k = R$. Is this model useful for recommendations?
  A: No, therefore we need regularization

Likelihood of data

$$\mathbb{P}\left\{R \mid P, Q, \sigma^2\right\} = \prod_{u \in U} \prod_{i \in I} \left[\mathcal{N}\left(r_{ui} \mid \mathbf{p}_u \mathbf{q}_i, \sigma^2\right)\right]^{I_{ui}},$$

where $I_{ui} = \mathbb{1}\{r_{ui} \neq 0\}$

Probabilistic Matrix Factorization [12]

Likelihood of data

$$\mathbb{P}\left\{R \mid P, Q, \sigma^2\right\} = \prod_{u \in U} \prod_{i \in I} \left[\mathcal{N}\left(r_{ui} \mid \mathbf{p}_u \mathbf{q}_i, \sigma^2\right)\right]^{I_{ui}},$$

where $I_{ui} = \mathbb{1}\{r_{ui} \neq 0\}$

Priors

$$\mathbb{P}\left\{P \mid \sigma_P^2\right\} = \prod_{u \in U} \mathcal{N}\left(\mathbf{p}_u \mid 0, \sigma_P^2 \mathbf{I}\right) \quad \mathbb{P}\left\{Q \mid \sigma_Q^2\right\} = \prod_{i \in I} \mathcal{N}\left(\mathbf{q}_i \mid 0, \sigma_Q^2 \mathbf{I}\right)$$

Likelihood of data

$$\mathbb{P}\left\{R \mid P, Q, \sigma^2\right\} = \prod_{u \in U} \prod_{i \in I} \left[\mathcal{N}\left(r_{ui} \mid \mathbf{p}_u \mathbf{q}_i, \sigma^2\right)\right]^{I_{ui}},$$

where $I_{ui} = \mathbb{1}\{r_{ui} \neq 0\}$

Priors

$$\mathbb{P}\left\{P \mid \sigma_P^2\right\} = \prod_{u \in U} \mathcal{N}\left(\mathbf{p}_u \mid 0, \sigma_P^2 \mathbf{I}\right) \quad \mathbb{P}\left\{Q \mid \sigma_Q^2\right\} = \prod_{i \in I} \mathcal{N}\left(\mathbf{q}_i \mid 0, \sigma_Q^2 \mathbf{I}\right)$$

Posterior

$$\ln \mathbb{P}\left\{P, Q \mid R, \sigma^2, \sigma_P^2, \sigma_Q^2\right\} \to \max \quad \Longleftrightarrow$$

$$\mathcal{L} = \underbrace{\frac{1}{2} \sum_{u \in U, i \in I} I_{ui} (r_{ui} - \mathbf{p}_u \mathbf{q}_i)^2}_{\text{sum over observer } r_{ui}} + \underbrace{\frac{\lambda_P}{2} \sum_{u \in U} \|\mathbf{p}_u\|_F^2}_{L_2 \text{ reg}} + \underbrace{\frac{\lambda_Q}{2} \sum_{i \in I} \|\mathbf{q}_i\|_F^2}_{L_2 \text{ reg}} \to \min$$

Recall FM model

$$\phi_{FM}(\mathbf{w}, \mathbf{x}) = \phi_{LM}(\mathbf{w}, \mathbf{x}) + \sum_{j_1=1}^{n} \sum_{j_2=j_1+1}^{n} (\mathbf{w}_{j_1} \cdot \mathbf{w}_{j_2}) x_{j_1} x_{j_2},$$

where $\mathbf{x} \in \mathbb{R}^n$ and loss function $\mathcal{L} = \frac{1}{2}(y - \phi_{FM}(\mathbf{w}, \mathbf{x}))^2$

Recall FM model

$$\phi_{FM}(\mathbf{w}, \mathbf{x}) = \phi_{LM}(\mathbf{w}, \mathbf{x}) + \sum_{j_1=1}^{n} \sum_{j_2=j_1+1}^{n} (\mathbf{w}_{j_1} \cdot \mathbf{w}_{j_2}) x_{j_1} x_{j_2},$$

where $\mathbf{x} \in \mathbb{R}^n$ and loss function $\mathcal{L} = \frac{1}{2}(y - \phi_{FM}(\mathbf{w}, \mathbf{x}))^2$

Assume that $\mathbf{x} = [\mathbf{u}, \mathbf{i}]$, where

- $\mathbf{u}$ — one-hot vector for user
- $\mathbf{i}$ — one-hot vector for item

Recall FM model

$$\phi_{FM}(\mathbf{w}, \mathbf{x}) = \phi_{LM}(\mathbf{w}, \mathbf{x}) + \sum_{j_1=1}^{n} \sum_{j_2=j_1+1}^{n} (\mathbf{w}_{j_1} \cdot \mathbf{w}_{j_2}) x_{j_1} x_{j_2},$$

where $\mathbf{x} \in \mathbb{R}^n$ and loss function $\mathcal{L} = \frac{1}{2}(y - \phi_{FM}(\mathbf{w}, \mathbf{x}))^2$

Assume that $\mathbf{x} = [\mathbf{u}, \mathbf{i}]$, where

- $\mathbf{u}$ — one-hot vector for user
- $\mathbf{i}$ — one-hot vector for item

Therefore, if we *skip LM term*(**Q:** what if not?) we will get

$$\phi_{FM}(P, Q, \mathbf{x}) = \sum_{u=1}^{|U|} \sum_{i=1}^{|I|} \mathbb{1}\{\mathbf{u}_u = 1 \wedge \mathbf{i}_i = 1\} \mathbf{p}_u \cdot \mathbf{q}_i$$

Recall FM model

$$\phi_{FM}(\mathbf{w}, \mathbf{x}) = \phi_{LM}(\mathbf{w}, \mathbf{x}) + \sum_{j_1=1}^{n} \sum_{j_2=j_1+1}^{n} (\mathbf{w}_{j_1} \cdot \mathbf{w}_{j_2}) x_{j_1} x_{j_2},$$

where $\mathbf{x} \in \mathbb{R}^n$ and loss function $\mathcal{L} = \frac{1}{2}(y - \phi_{FM}(\mathbf{w}, \mathbf{x}))^2$

Assume that $\mathbf{x} = [\mathbf{u}, \mathbf{i}]$, where

- $\mathbf{u}$ — one-hot vector for user
- $\mathbf{i}$ — one-hot vector for item

Therefore, if we *skip LM term*(**Q:** what if not?) we will get

$$\phi_{FM}(P, Q, \mathbf{x}) = \sum_{u=1}^{|U|} \sum_{i=1}^{|I|} \mathbb{1}\{\mathbf{u}_u = 1 \wedge \mathbf{i}_i = 1\} \mathbf{p}_u \cdot \mathbf{q}_i$$

If we plug it in sqare-loss and add L_2 reg. we will obtain exactly loss for PMF

PMF via SGD

- Slow to train when number of observations is very large, therefore
- Slow convergence rate

Alternating Least Squares (ALS) [9]

Let $\mathcal{K}$ be a set of (u, i) pairs for which r_{ui} is known, therefore

$$\mathcal{L} = \frac{1}{2} \sum_{(u,i) \in \mathcal{K}} (r_{ui} - \mathbf{p}_u \mathbf{q}_i)^2 + \lambda(\|\mathbf{p}_u\|_F^2 + \|\mathbf{q}_i\|_F^2)$$

Optimization of $\mathcal{L}$ can be done more efficiently due to the following

Observation

If during optimization we fix P in $\mathcal{L}$ then optimization problem becomes convex wrt to Q and vice versa. Therefore can be solved in closed-form

Faster than SGD convergence

Implicit ALS (iALS) [9, 8]

- $p_{ui} = \mathbb{1}\{r_{ui} > 0\}$ — implicit feedback (observed interaction)

Implicit ALS (iALS) [9, 8]

- $p_{ui} = \mathbb{1}\{r_{ui} > 0\}$ — implicit feedback (observed interaction)
- If $p_{ui} = 0$ our beliefs are associated with varying *confidence* levels

Implicit ALS (iALS) [9, 8]

- $p_{ui} = \mathbb{1}\{r_{ui} > 0\}$ — implicit feedback (observed interaction)
- If $p_{ui} = 0$ our beliefs are associated with varying *confidence* levels
- Zero values of p_{ui} are associated with low confidence

Implicit ALS (iALS) [9, 8]

- $p_{ui} = \mathbb{1}\{r_{ui} > 0\}$ — implicit feedback (observed interaction)
- If $p_{ui} = 0$ our beliefs are associated with varying *confidence* levels
- Zero values of p_{ui} are associated with low confidence
- As r_{ui} grows we have a stronger indication that u indeed likes i

Implicit ALS (iALS) [9, 8]

- $p_{ui} = \mathbb{1}\{r_{ui} > 0\}$ — implicit feedback (observed interaction)
- If $p_{ui} = 0$ our beliefs are associated with varying *confidence* levels
- Zero values of p_{ui} are associated with low confidence
- As r_{ui} grows we have a stronger indication that u indeed likes i

Popular choices for confidence c_{ui}

- $1 + \alpha r_{ui}$
- $1 + \alpha \log(1 + r_{ui}/\epsilon)$

Implicit ALS (iALS) [9, 8]

- $p_{ui} = \mathbb{1}\{r_{ui} > 0\}$ — implicit feedback (observed interaction)
- If $p_{ui} = 0$ our beliefs are associated with varying *confidence* levels
- Zero values of p_{ui} are associated with low confidence
- As r_{ui} grows we have a stronger indication that u indeed likes i

Popular choices for confidence c_{ui}

- $1 + \alpha r_{ui}$
- $1 + \alpha \log(1 + r_{ui}/\epsilon)$

We obtain following loss

$$\mathcal{L} = \frac{1}{2} \sum_{(u,i) \in U \times I} c_{ui}(p_{ui} - \mathbf{p}_u \mathbf{q}_i)^2 + \frac{\lambda_P}{2} \sum_{u \in U} \|\mathbf{p}_u\|_F^2 + \frac{\lambda_Q}{2} \sum_{i \in I} \|\mathbf{q}_i\|_F^2$$

Can be solved via ALS (see [9] for details)

Large Scale MF [5]

Q: How do we usually scale SGD?

Large Scale MF [5]

Q: How do we usually scale SGD?
A: Parameter Server, Data/Model Parallelism

Large Scale MF [5]

Q: How do we usually scale SGD?
A: Parameter Server, Data/Model Parallelism

Q: How can we apply them to SGD MF?

Large Scale MF [5]

Q: How do we usually scale SGD?
A: Parameter Server, Data/Model Parallelism

Q: How can we apply them to SGD MF?
A:

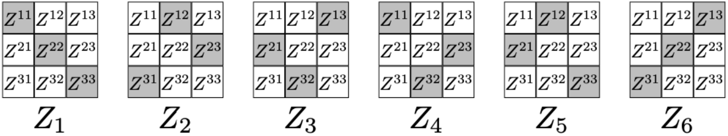


Рис.: Blocks in grey can be processed independently. Z_i — i-th sub-epoch. $Z_1, \ldots, Z_6$ — single epoch

Table of Contents

1. Evaluation

2. Matrix Completion

3. Link Prediction
   - Formulation
   - Large Scale Link Prediction

4. Session-based Recommendations

5. Learning to Rank

6. Resume

7. RecSys at VK

Link Prediction

Aggregated data:

- $G = \langle U \cup I, E \rangle, w \colon E \to \mathbb{R}$ — weighted bipartite graph, where
- $E = \{(u, i) \mid \exists t \colon (u_t, i_t, y_t) \in D \wedge u_t = u \wedge i_t = i\} \subset U \times I$
- $w(u, i) = r_{ui} = agg\{(u_t, i_t, y_t) \in D \mid u_t = u \wedge i_t = i\}$

Link Prediction

Aggregated data:

- $G = \langle U \cup I, E \rangle, w \colon E \to \mathbb{R}$ — weighted bipartite graph, where
- $E = \{(u,i) \mid \exists t \colon (u_t, i_t, y_t) \in D \wedge u_t = u \wedge i_t = i\} \subset U \times I$
- $w(u,i) = r_{ui} = agg\{(u_t, i_t, y_t) \in D \mid u_t = u \wedge i_t = i\}$

Task:

- Predict weight $w(u,i)$ of non-existing edge (u,i)

In more general case G is a Heterogeneous Information Network (HIN)

Definition (Heterogeneous Information Network)

$G = \langle V, E, \Phi, \Psi, w \rangle$ is a Heterogeneous Information Network (HIN), where $\Phi\colon V \to A$ — mapping from vertex to its type, $\Psi\colon E \to X$ — mapping from edge to its type, such that $|A| > 1$ or $|X| > 1$.

In more general case G is a Heterogeneous Information Network (HIN)

Definition (Heterogeneous Information Network)

$G = \langle V, E, \Phi, \Psi, w \rangle$ is a Heterogeneous Information Network (HIN), where $\Phi\colon V \to A$ — mapping from vertex to its type, $\Psi\colon E \to X$ — mapping from edge to its type, such that $|A| > 1$ or $|X| > 1$.

HIN example (movies recommendations):

- $V = Actors \cup Movies \cup Users \cup Genres$, verticies types $A = \{actor, movie, user, genre\}$
- Edges types $X = \{starred\_in, rated, belongs\_to\}$

- DeepWalk [13], Node2Vec [6]
- Graph Representation Learning (GCMC [2])
- HINs (metapath2vec [3], HIN2vec [4])

Large Scale LP

- Pytorch BigGraph [10]
- GraphVite [21]

Table of Contents

1. Evaluation

2. Matrix Completion

3. Link Prediction

4. Session-based Recommendations

5. Learning to Rank

6. Resume

7. RecSys at VK

Aggregated data:

- $S_u = \langle (i_1, y_1), (i_2, y_2), \ldots, (i_{n_u}, y_{n_u}) \rangle$ — u-th user session chronologically ordered

Aggregated data:

- $S_u = \langle (i_1, y_1), (i_2, y_2), \ldots, (i_{n_u}, y_{n_u}) \rangle$ — u-th user session chronologically ordered

Task:

- $P(i_{n_u+1} = i \mid S_u)$ — probability of the next event given user session S_u

- RNN based [18], BERT2Rec [17]

- RNN based [18], BERT2Rec [17]

Remark

But be careful! Most neural achitectures can not outperform simple solutions. See [11].

Table of Contents

1. Evaluation

2. Matrix Completion

3. Link Prediction

4. Session-based Recommendations

5. Learning to Rank

6. Resume

7. RecSys at VK

Learning to Rank

Table of Contents

1. Evaluation

2. Matrix Completion

3. Link Prediction

4. Session-based Recommendations

5. Learning to Rank

6. Resume

7. RecSys at VK

Resume

1. There are many ways to formulate RecSys problem as ML problem
   - Matrix Completion
   - Link Prediction
   - Session-based
   - Learning to Rank
2. Additionaly each formulation can be foother adjusted to deal with different kinds of user feedback
   - Explicit
   - Implicit
3. Evaluation (Protocols, Metrics)

Table of Contents

1. Evaluation

2. Matrix Completion

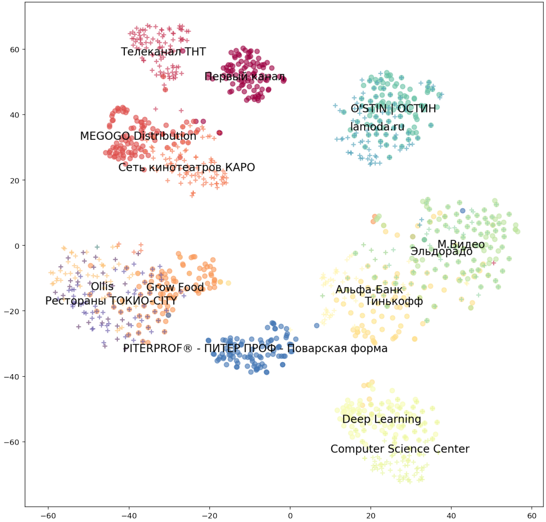
3. Link Prediction

4. Session-based Recommendations

5. Learning to Rank

6. Resume

7. RecSys at VK
   - Similar Communities and Domains Search
   - Тематические Ленты


Телеканал ТНТ
Первый канал
O'STIN | ОСТИН
lamoda.ru
MEGOGO Distribution
Сеть кинотеатров КАРО
М.Видео
Эльдорадо
Ollis
Grow Food
Рестораны ТОКИО-CITY
Альфа-Банк
Тинькофф
PITERPROF® - ПИТЕР ПРОФ - Поварская форма
Deep Learning
Computer Science Center
60
40
20
0
−20
−40
−60
−60
−40
−20
0
20
40
60

Тематические Ленты[2]

[2] https://vk.com/video-187376020_456239017

References I

[1] O. Barkan, N. Koenigstein, E. Yogev, and O. Katz. Cb2cf: a neural multiview content-to-collaborative filtering model for completely cold item recommendations. In *Proceedings of the 13th ACM Conference on Recommender Systems*, pages 228–236, 2019.

[2] R. v. d. Berg, T. N. Kipf, and M. Welling. Graph convolutional matrix completion. *arXiv preprint arXiv:1706.02263*, 2017.

[3] Y. Dong, N. V. Chawla, and A. Swami. metapath2vec: Scalable representation learning for heterogeneous networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 135–144, 2017.

[4] T.-y. Fu, W.-C. Lee, and Z. Lei. Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1797–1806, 2017.

[5] R. Gemulla, E. Nijkamp, P. J. Haas, and Y. Sismanis. Large-scale matrix factorization with distributed stochastic gradient descent. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 69–77, 2011.

[6] A. Grover and J. Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864, 2016.

[7] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*, pages 173–182, 2017.

[8] X. He, H. Zhang, M.-Y. Kan, and T.-S. Chua. Fast matrix factorization for online recommendation with implicit feedback. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 549–558, 2016.

[9] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *2008 Eighth IEEE International Conference on Data Mining*, pages 263–272. Ieee, 2008.

[10] A. Lerer, L. Wu, J. Shen, T. Lacroix, L. Wehrstedt, A. Bose, and A. Peysakhovich. Pytorch-biggraph: A large-scale graph embedding system. *arXiv preprint arXiv:1903.12287*, 2019.

[11] M. Ludewig, N. Mauro, S. Latifi, and D. Jannach. Performance comparison of neural and non-neural approaches to session-based recommendation. In *Proceedings of the 13th ACM Conference on Recommender Systems*, pages 462–466, 2019.

[12] A. Mnih and R. R. Salakhutdinov. Probabilistic matrix factorization. In *Advances in neural information processing systems*, pages 1257–1264, 2008.

[13] B. Perozzi, R. Al-Rfou, and S. Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710, 2014.

[14] S. Rendle. Factorization machines. In *2010 IEEE International Conference on Data Mining*, pages 995–1000. IEEE, 2010.

[15] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618*, 2012.

[16] F. Ricci, L. Rokach, and B. Shapira. Introduction to recommender systems handbook. In *Recommender systems handbook*, pages 1–35. Springer, 2011.

[17] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, and P. Jiang. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 1441–1450, 2019.

[18] Y. K. Tan, X. Xu, and Y. Liu. Improved recurrent neural networks for session-based recommendations. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, pages 17–22, 2016.

[19] A. Van den Oord, S. Dieleman, and B. Schrauwen. Deep content-based music recommendation. In *Advances in neural information processing systems*, pages 2643–2651, 2013.

[20] J. Weston, S. Bengio, and N. Usunier. Wsabie: Scaling up to large vocabulary image annotation. In *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

[21] Z. Zhu, S. Xu, J. Tang, and M. Qu. Graphvite: A high-performance cpu-gpu hybrid system for node embedding. In *The World Wide Web Conference*, pages 2494–2504, 2019.