Fundamentals of Deep Recommender Systems

Wenqi Fan
The Hong Kong Polytechnic University
https://wenqifan03.github.io, wenqifan@polyu.edu.hk

Tutorial website: https://advanced-recommender-systems.github.io/ijcai2021-tutorial/
A General Architecture of Deep Recommender System

Embedding layer

Prediction layer

Hidden layers (e.g., MLP, CNN, RNN, etc.)

User
Item
Context
Interaction

Field 1
Field m
Field M

Hidden layers (e.g., MLP, CNN, RNN, etc.)
NeuMF unifies the strengths of MF and MLP in modeling user-item interactions.

- **MF** uses an inner product as the interaction function
- **MLP** is more sufficient to capture the complex structure of user interaction data
The **wide linear models** can memorize seen feature interactions using cross-product feature transformations.

The **deep models** can generalize to previously unseen feature interactions through low-dimensional embeddings.
Neural **Factorization Machines** (NFMs) “deepens” FM by placing hidden layers above second-order **feature interaction** modeling.
Neural FM

Neural Factorization Machines (NFMs) “deepens” FM by placing hidden layers above second-order feature interaction modeling.

“Deep layers” learn higher-order feature interactions only, being much easier to train.

Bilinear Interaction Pooling:

\[
f_{BI}(V_x) = \sum_{i=1}^{n} \sum_{j=i+1}^{n} x_i v_i \odot x_j v_j
\]
DeepFM ensembles FM and DNN and to low- and high-order feature interactions simultaneously from the input raw features.

Prediction Model: \( \hat{y} = \text{sigmoid}(y_{FM} + y_{DNN}) \)
DeepFM ensembles FM and DNN and to low- and high-order feature interactions simultaneously from the input raw features.

**FM component (low-order)**

\[ y_{FM} = \langle w, x \rangle + \sum_{j_1=1}^{d} \sum_{j_2=j_1+1}^{d} \langle V_{i}, V_{j} \rangle x_{j_1} \cdot x_{j_2} \]

**Deep component (high-order)**

\[ a^{(l)} = [e_1, e_2, \ldots, e_m] \]
\[ a^{(l+1)} = \sigma(W^{(l)}a^{(l)} + b^{(l)}) \]
\[ y_{DNN} = \sigma(W^{H|+1} \cdot a^{H} + b^{H|+1}) \]

Prediction Model: \[ \hat{y} = \text{sigmoid}(y_{FM} + y_{DNN}) \]
Collaborative Filtering with users’ social relations
(Social Recommendation)

US could see millions of coronavirus cases and 100,000 or more deaths

Dr. Anthony Fauci

News
Collaborative Filtering with users’ social relations
(Social Recommendation)

Users might be affected by direct/distant neighbors.
- Information diffusion
- Users with high reputations

US could see millions of coronavirus cases and 100,000 or more deaths
Collaborative Filtering with users’ social relations
(Social Recommendation)

Users might be affected by direct/distant neighbors.
- Information diffusion
- Users with high reputations

Bi-LSTM with attention mechanisms

Social Sequences via Random Walk techniques

Deep Social Collaborative Filtering, RecSys, 2019
DASO

Collaborative Filtering with users’ social relations (Social Recommendation)

- User behave and interact **differently** in the item/social domains.
Collaborative Filtering with users’ social relations
(Social Recommendation)

- User behave and interact differently in the item/social domains.

- Learning separated user representations in two domains.
Collaborative Filtering with users’ social relations
(Social Recommendation)

- **User behave and interact differently** in the item/social domains.

- **Learning separated user representations in two domains.**

**Bidirectional Knowledge Transfer with Cycle Reconstruction**

\[ \mathbf{p}_i^I \rightarrow h^{I \rightarrow S}(\mathbf{p}_i^I) \rightarrow h^{S \rightarrow I}(h^{I \rightarrow S}(\mathbf{p}_i^I)) \approx \mathbf{p}_i^I \]

\[ \mathcal{L}_{cyc}(h^{S \rightarrow I}, h^{I \rightarrow S}) = \sum_{i=1}^{N} (\|h^{S \rightarrow I}(h^{I \rightarrow S}(\mathbf{p}_i^I)) - \mathbf{p}_i^I\|_2 + \|h^{I \rightarrow S}(h^{S \rightarrow I}(\mathbf{p}_i^S)) - \mathbf{p}_i^S\|_2) \]

Deep Adversarial Social Recommendation, IJCAI, 2019
Optimization for Ranking Tasks

- Negative Sampling’s Main Issue:
  - It often generates low-quality negative samples that do not help you learn good representation.
Optimization for Ranking Tasks

- **Negative Sampling’s Main Issue:**
  - It often generates low-quality negative samples that do not help you learn good representation \[\text{Cai and Wang, 2018; Wang et al., 2018b}\].

Deep Adversarial Social Recommendation, IJCAI, 2019
Optimization for Ranking Tasks

- **Negative Sampling’s Main Issue:**
  - It often generates low-quality negative samples that do not help you learn good representation [Cai and Wang, 2018; Wang et al., 2018b].

Deep Adversarial Social Recommendation, IJCAI, 2019
DASO

Item Domain Adversarial Learning

Cyclic User Modeling

Social Domain Adversarial Learning

- Item Domain Representations for Generator
- User Representations on Item Domain after Mapping (S->I)
- Item Domain Representations for Discriminator

- Social Domain Representations for Generator
- User Representations on Social Domain after Mapping (I->S)
- Social Domain Representations for Discriminator

User-Item Interactions

Real Samples

Loss/Reward

\( f_{\phi_0}^{I} (x', y') \)

Generated Samples

\( g_{\phi_G}^{I} (p_{SI}, q') \)

\( p(v|u) \)

Discriminator

Generator

Reward

User Representations on Social Domain after Mapping (I->S)

User Representations on Item Domain after Mapping (S->I)

User-User Connections

Real Samples

Loss/Reward

\( f_{\phi_0}^{S} (x^S, x^S_k) \)

\( g_{\phi_G}^{S} (p_{IS}, p_{IS_k}^S) \)

\( p(u_k|u) \)

Discriminator

Generator

Generated Samples

User-User Connections

Real Samples

Deep Adversarial Social Recommendation, IJCAI, 2019
Deep Adversarial Social Recommendation, IJCAI, 2019
**Item Domain Discriminator Model**

- **Discriminator**
  - **Goal:** distinguish real user-item pairs (i.e., real samples) and the generated “fake” samples (relevant)

\[
D^I(u_i, v_j; \phi_D^I) = \sigma(f_{\phi_D^I}(x_i^I, y_j^I)) = \frac{1}{1 + \exp(-f_{\phi_D^I}(x_i^I, y_j^I))} \quad \text{(Sigmoid)}
\]

**Score function:**

\[
f_{\phi_D^I}(x_i^I, y_j^I) = (x_i^I)^T y_j^I + a_j,
\]

Deep Adversarial Social Recommendation, IJCAI, 2019
Item Domain Generator Model

Generator Model

Goal:
1. Approximate the underlying real conditional distribution $\mathbf{p}_i^{\text{real}}(\mathbf{v}|\mathbf{u}_i)$
2. Generate (select/sample) the most relevant items for any given user $\mathbf{u}_i$.

$$G^I(v_j|u_i; \theta_G^I) = \frac{\exp(g_{\theta_G^I}^I(p_{SI}^I, q_j^I))}{\sum_{v_j \in V} \exp(g_{\theta_G^I}^I(p_{SI}^I, q_j^I))}$$

$$g_{\theta_G^I}^I(p_{SI}^I, q_j^I) = (p_{SI}^I)^T q_j^I + b_j$$

Optimization with Policy Gradient
Sequential (Session-based) Recommendation

Session-based Recommendations with Recurrent Neural Networks, ICLR, 2016.
BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer, CIKM, 2019.
Sequential (Session-based) Recommendation

Session-based Recommendations with Recurrent Neural Networks, ICLR, 2016.
BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer, CIKM, 2019.
Sequential (Session-based) Recommendation

GRU based sequential recommendation method (GRU4Rec)

Next Item

BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer, CIKM, 2019.

Session-based Recommendations with Recurrent Neural Networks, ICLR, 2016.
Shortcomings of Existing Deep Recommender Systems

Recommendation Policies
- Offline optimization
- Short-term reward
Shortcomings of Existing Deep Recommender Systems

Recommendation Policies
- Offline optimization
- Short-term reward

Graph-structured Data
- Information isolated island Issue: ignore implicit/explicit relationships among instances
Shortcomings of Existing Deep Recommender Systems

Recommendation Policies
- Offline optimization
- Short-term reward

Graph-structured Data
- Information isolated island issue: ignore implicit/explicit relationships among instances

Manually Designed Architectures
- Expert knowledge
- Time and engineering efforts
Shortcomings of Existing Deep Recommender Systems

**Recommendation Policies**
- Offline optimization
- Short-term reward

**Graph-structured Data**
- Information isolated island issue: ignore implicit/explicit relationships among instances

**Manually Designed Architectures**
- Expert knowledge
- Time and engineering efforts

**Poisoning attacks:**
- Promote/demote items
- White/grey/black-box attacks