Outline

Morning program
- Preliminaries
- Modeling user behavior
- Semantic matching
- Learning to rank

Afternoon program
- Entities
- Generating responses
- Recommender systems
- Industry insights
- Q & A

Outline

Morning program
- Preliminaries
- Modeling user behavior
- Semantic matching
- Learning to rank

Afternoon program
- Entities
- Generating responses
- Recommender systems
  - Items and Users
  - Matrix factorization
  - Matrix factorization as a network
  - Side information
  - Richer models
  - Other tasks
  - Wrap-up
- Industry insights
- Q & A

Recommender systems – The task

- Build a model that estimates how a user will like an item.
- A typical recommendation setup has
  - matrix with users and items
  - plus ratings of users for items reflecting past/known preferences

  and tries to predict future preferences
- This is **not** about rating prediction

2			4	5	
5		4			1
		5		2	
	1		5		4
		4		2	
4	5		1		

[Karatzoglou and Hidasi, Deep Learning for Recommender Systems, RecSys '17, 2017]

Approaches to recommender systems

- Collaborative filtering
  - Based on analyzing users' behavior and preferences such as ratings given to movies or books
- Content-based filtering
  - Based on matching the descriptions of items and users' profiles
  - Users' profiles are typically constructed using their previous purchases/ratings, their submitted queries to search engines and so on
- A hybrid approach

Warm, cold

- Cold start problem
  - **User** cold-start problem – generate recommendations for a new user / a user for whom very few preferences are known
  - **Item** cold-start problem – recommendation items that are new / for which very users have shared ratings or preferences
- Cold items/users
- Warm items/users

Outline

Morning program
- Preliminaries
- Modeling user behavior
- Semantic matching
- Learning to rank

Afternoon program
- Entities
- Generating responses
- Recommender systems
  - Items and Users
  - Matrix factorization
  - Matrix factorization as a network
  - Side information
  - Richer models
  - Other tasks
  - Wrap-up
- Industry insights
- Q & A

Matrix factorization

- The recommender system's work horse

Matrix factorization

- Discover the latent features underlying the interactions between users and items
- Don't rely on imputation to fill in missing ratings and make matrix dense
- Instead, model observed ratings directly, avoid overfitting through a regularized model
- Minimize the regularized squared error on the set of known ratings:

$$\min_{u,v} \sum_{i,j \in R} (r_{i,j} - u_i^T v_j) + \lambda(\|u_i\|^2 + \|v_j\|^2)$$

Popular methods for minimizing include **stochastic gradient descent** and **alternating least squares**

Outline

Morning program
- Preliminaries
- Modeling user behavior
- Semantic matching
- Learning to rank

Afternoon program
- Entities
- Generating responses
- Recommender systems
  - Items and Users
  - Matrix factorization
  - Matrix factorization as a network
  - Side information
  - Richer models
  - Other tasks
  - Wrap-up
- Industry insights
- Q & A

A feed-forward neural network view

[Raimon and Basilico, Deep Learning for Recommender Systems, 2017]

A deeper view

Matrix factorization vs. feed-forward network

- Two models are very similar
  - Embeddings, MSE loss, gradient-based optimization
- Feed-forward net can learn different embedding combinations than a dot product
- Capturing pairwise interactions through feed-forward net requires a huge amount of data
- This approach is not superior to properly tuned traditional matrix factorization approach

Great escape . . .

- Side information
- Richer models
- Other tasks

Outline

Morning program
- Preliminaries
- Modeling user behavior
- Semantic matching
- Learning to rank

Afternoon program
- Entities
- Generating responses
- Recommender systems
  - Items and Users
  - Matrix factorization
  - Matrix factorization as a network
  - Side information
  - Richer models
  - Other tasks
  - Wrap-up
- Industry insights
- Q & A

Side information for recommendation

(1)

(2)

(3)

(4)

Side information for recommendation

- Textual side information
  - Product description, reviews, etc.
  - Extraction: RNNs, one dimensional CNNs, word embeddings, paragraph vectors
  - Applications: news, products, books, publication
- Images
  - Product pictures, video thumbnails
  - Extraction: CNNs
  - Applications: fashion, video
- Music/audio
  - Extraction: CNNs and RNNs
  - Applications: music

Textual side information

- Content2vec [Nedelec et al., 2016]
- Using associated textual information for recommendations [Bansal et al., 2016]

Textual information for improving recommendations

- Task: paper recommendation
- Item representation
  - Text representation: RNN based
  - Item-specific embeddings created using MF
  - Final representation: item + text embeddings

Images in recommendation

Visual Bayesian Personalized Ranking (BPR) [He and McAuley, 2016]

- Bias terms
- MF model
- Visual part:
  - Pretrained CNN features
  - Dimension reduction through embeddings
- BPR loss

Outline

Morning program
- Preliminaries
- Modeling user behavior
- Semantic matching
- Learning to rank

Afternoon program
- Entities
- Generating responses
- Recommender systems
  - Items and Users
  - Matrix factorization
  - Matrix factorization as a network
  - Side information
  - Richer models
  - Other tasks
  - Wrap-up
- Industry insights
- Q & A

Alternative models

- Restricted Boltzman Machines [Salakhutdinov et al., 2007]
- Auto-encoders [Wu et al., 2016]
- Prod2vec [Grbovic et al., 2015]
- Wide + Deep models [Cheng et al., 2016]

Restricted Boltzman Machines – RBM

- Generative stochastic neural network
- Visible and hidden units connected by weights
- Activation probabilities:
  $p(h_j = 1|v) = \sigma(b_j^h + \sum_{i=1}^m w_{i,j} v_i)$
  $p(v_i = 1|h) = \sigma(b_i^v + \sum_{j=1}^n w_{i,j} h_j)$
- Training
  - Set visible units based on data, sample hidden units, then sample visible units
  - Modify weights to approach the configuration of visible units to the data
- In recommendation:
  - Visible units: ratings on the movie
  - Vector of length 5 (for each rating value) in each unit
  - Units corresponding to users who not rated the movie are ignored

Auto-encoders

Auto-encoders

- One hidden layer
- Same number of input and output units
- Try to reconstruct the input on the output
- Hidden layer: compressed representation of the data

Constraining the model: improve generalization

- Sparse auto-encoders: activation of units are limited
- Denoising auto-encoders: corrupt the input

Auto-encoders for recommendation

Reconstruct corrupted user interaction vectors [Wu et al., 2016]

- Collaborative Denoising Auto-Encoder (CDAE)
- The link between nodes are associated with different weights
- The links with red color are user specific
- Other weights are shared across all the users

Prod2vec and Item2vec

- Prod2vec and item2vec: Item-item co-occurrence factorization
- User2vec: User-user co-occurrence factorization
- The two approaches can be combined [Liang et al., 2016]

user2vec
(Continuous Bag of Words)

Wide + Deep models

- Combination of two models
- Deep neural network
  - On embedded item features
  - In charge of generalization
- Linear model
  - On embedded item feature
  - And cross product of item features
  - In charge of memorization on binarized features
- [Cheng et al., 2016]

Outline

Morning program
- Preliminaries
- Modeling user behavior
- Semantic matching
- Learning to rank

Afternoon program
- Entities
- Generating responses
- Recommender systems
  - Items and Users
  - Matrix factorization
  - Matrix factorization as a network
  - Side information
  - Richer models
  - Other tasks
  - Wrap-up
- Industry insights
- Q & A

Other tasks

- Session-based recommendation
- Contextual sequence prediction
- Time-sensitive sequence prediction

- Causality in recommendations
- Recommendation as question answering
- Deep reinforcement learning for recommendations

Session-based recommendation

- Treat recommendations as a sequence classification problem
- Input: a sequence of user actions (purchases/ratings of items)
- Output: next action
- Disjoint sessions (instead of consistent user history)

GRU4Rec

Network structure [Hidasi et al., 2016]

- Input: one hot encoded item ID
- Output: scores over all items
- Goal: predicting the next item in the session

Adapting GRU to session-based recommendations

- Session-parallel mini-batching: to handle sessions of (very) different length and lots of short sessions
- Sampling on the output: to handle lots of items (inputs,outputs)

GRU4Rec

Session-parallel mini-batches

- Mini-batch is defined over sessions

Output sampling

- Computing scores for all items (100K 1M) in every step is slow
- One positive item (target) + several samples
- Fast solution: scores on mini-batch targets
- Items of the other mini-batches are negative samples for the current mini-batch

Contextual sequence prediction

- Input: sequence of contextual user actions, plus current context
- Output: probability of next action
- E.g. "Given all the actions a user has taken so far, what's the most likely video they're going to play right now?" [Beutel et al., 2018]

Time-sensitive sequence prediction

- Recommendations are actions at a moment in time
  - Proper modeling of time and system dynamics is critical
- Experiment on a Netflix internal dataset
  - Context:
    - Discrete time – Day-of-week: Sunday, Monday, . . . Hour-of-day
    - Continuous time (Timestamp)
  - Predict next play (temporal split data)

[Raimon and Basilico, Deep Learning for Recommender Systems, 2017]

Causality in recommendations – [Schnabel et al., 2016]

- Virtually all data for training recommender systems is subject to selection biases
  - In movie recommendation, users typically watch and rate movies they like, rarely movies they do not like
- View recommendation from causal inference perspective – exposing user to item is intervention analogous to exposing patient to treatment in medical study
- Propensity-weighted MF method – propensities act as weights on loss terms

$$\min_{u,v} \sum_{i,j \in R} \frac{1}{P_{i,j}} (r_{i,j} - u_i^T v_j) + \lambda(\|u_i\|^2 + \|v_j\|^2)$$

- Performance: MF vs. propensity weighted MF

(As $\alpha \to 0$, data is increasingly missing not at random, only revealing top rated items.)

- How to incorporate this in neural networks for recommender systems?

Recommendations as question answering – [Dodge et al., 2015]

- Conversational recommendation agent
  - (1) question-answering (QA), (2) recommendation, (3) mix of recommendation and QA and (4) general dialog about the topic (chit-chat)

Task 2: Recommendation

Schindler's List, The Fugitive, Apocalypse Now, Pulp Fiction, and The Godfather are films I really liked.
Can you suggest a film? The Hunt for Red October

Some movies I like are Heat, Kids, Fight Club, Shaun of the Dead, The Avengers, Skyfall, and Jurassic Park.
Can you suggest something else I might like? Ocean's Eleven

Task 3: QA + Recommendation Dialog

I loved Billy Madison, My Neighbor Totoro, Blades of Glory, Bio-Dome, Clue, and Happy Gilmore.
I'm looking for a Music movie. School of Rock
What else is that about? Music, Musical, Jack Black, school, teacher, Richard Linklater, rock, guitar
I like rock and roll movies more. Do you know anything else? Little Richard

Tombstone, Legends of the Fall, Braveheart, The Net, Outbreak, and French Kiss are films I really liked.
I'm looking for a Fantasy movie. Jumanji
Who directed that? Joe Johnston
I like Tim Burton movies more. Do you know anything else? Big Fish

- Memory network jointly trained on all (four) tasks performs best
- Incorporate short and long term memory and can use local context and knowledge bases of facts
- Performance on QA needs a real boost
- Performance degraded rather than improved when training on all four tasks at once

Deep reinforcement learning for recommendations – [Zhao et al., 2017]

- MDP-based formulations of recommender systems go back to early 2000s
- Use of reinforcement learning has two advantages
  1. can continuously update strategies during interactions
  2. are able to learn strategy that maximizes the long-term cumulative reward from users
- List-wise recommendation framework, which can be applied in scenarios with large and dynamic item spaces
- Uses Actor-Critic network
- Integrate multiple orders – positional order, temporal order
- Needs proper evaluation in live environment

Outline

Morning program
- Preliminaries
- Modeling user behavior
- Semantic matching
- Learning to rank

Afternoon program
- Entities
- Generating responses
- Recommender systems
  - Items and Users
  - Matrix factorization
  - Matrix factorization as a network
  - Side information
  - Richer models
  - Other tasks
  - Wrap-up
- Industry insights
- Q & A

Wrap-up

- Current directions
  - Item/user embedding
  - Deep collaborative filtering
  - Feature extraction from content
  - Session- and context-based recommendation
  - Fairness, accuracy, confidentiality, and transparency (FACT)
- Deep learning can work well for recommendations
- Matrix factorization and deep learning are very similar in classic recommendation setup
- Lots of areas to explore

Resources

RankSys	:	`https://github.com/RankSys/RankSys`
LibRec	:	`https://www.librec.net`
LensKit	:	`http://lenskit.org`
LibMF	:	`https://www.csie.ntu.edu.tw/%7Ecjlin/libmf/`
proNet-core	:	`https://github.com/cnclabs/proNet-core`
rival	:	`http://rival.recommenders.net`
TensorRec	:	`https://github.com/jfkirk/tensorrec`