CSE 291:
Trends in Recommender Systems and Human Behavioral Modeling

Week 6 project proposals
Personalized Next Song Recommendation

Kiran Kannar, Rahul Dubey
Problem Statement

Given user song listening history, provide personalized next song recommendation using metric embeddings.
So far...

- Epoch 0: Music Recommendation
- Epoch 1: "There are known knowns"
  - Logistic Markov Embedding - Yes.com radio playlists
  - Personalized Ranking Metric Embeddings - POI recommendation (Foursquare, Gowalla)
  - Use of Now Playing dataset having user listening history
  - Made a distinction between playlists vs listening history
- Ah. Clarity!
- Proposed Extensions
- Looking at a BIG BIG dataset.
Dataset

NowPlaying: [http://dbis-nowplaying.uibk.ac.at/](http://dbis-nowplaying.uibk.ac.at/)

Our dataset, keeping users who listened to at least 50 songs
Total 9288 sessions
### Preliminary Results

#### PRME with k=20

<table>
<thead>
<tr>
<th>alpha</th>
<th>0.05</th>
<th>0.1</th>
<th>0.2</th>
<th>0.3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hit rate @ 50</td>
<td>0.2051</td>
<td>0.2026</td>
<td>0.2012</td>
<td>0.2019</td>
</tr>
<tr>
<td>MRR @ 50</td>
<td>0.0737</td>
<td>0.0730</td>
<td>0.0716</td>
<td>0.0711</td>
</tr>
</tbody>
</table>

#### PRME with alpha = 0.05

<table>
<thead>
<tr>
<th>K</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hit rate @ 50</td>
<td>0.1783</td>
<td>0.2051</td>
<td>0.2123</td>
<td>0.2182</td>
</tr>
<tr>
<td>MRR @ 50</td>
<td>0.0550</td>
<td>0.0737</td>
<td>0.0781</td>
<td>0.0805</td>
</tr>
</tbody>
</table>
Extensions & Avenues

1. Personalizing $\alpha_u$

$$D_{u,l^c,l} = \alpha_u D_{u,l}^P + (1 - \alpha_u) D_{l^c,l}^S$$

$$\theta := \theta + \gamma \frac{\partial}{\partial \theta} (\log \sigma(z) - \lambda \|\theta\|^2)$$

$$z = D_{u,l^c,l_j} - D_{u,l^c,l_i}$$

2. Friends of user

Adding a regularizer, $\|X(u) - X(f_u)\|^2$

Hypothesis testing of the use of social circles.

3. Using content-based features for coldstart

Extract tag and lyrics of a song, create its embedding and project these embeddings in PRME embedding space

4. Session based recommendation

PRME => session KNN => LCS and item KNN for recommendation
30Music Dataset, a collection of listening and playlists data retrieved from Internet radio stations through Last.fm API.

Courtesey: Turrin, R., Quadrana, M., Condorelli, A., Pagano, R., & Cremonesi, P. [30Music listening and playlists dataset](https://www.scale-labs.org/datasets/30music)

Note: We just got the dataset late last night! Now please give us 540 GB of RAM :) 

All experiments have been executed on Amazon AWS EC2 instances, creating a 9-node cluster with an aggregated power of 72 cores and 540GB RAM.
Thank you!
FashionGAN: A generative model for fashion recommendation

By Vignesh Gokul
Base paper

- Learning Visual Clothing Style with Heterogeneous Dyadic Co-occurrences (Andreas Veit and Balazs Kovacs and Sean Bell and Julian McAuley and Kavita Bala and Serge Belongie)

- The paper implements a Siamese CNN with strategic sampling to learn the embedding space for all items and use these embeddings to build a better item recommender system
Siamese CNN Architecture
Implementation

- Used VGG-16 (both untrained and pretrained)
- Batch size of 10
- Margin of 100
- Adam Optimizer
- Tensorflow
Extensions

- A generative model that could perform Image-Image mapping. (FashionGAN)

- A Siamese CNN to learn audio embeddings.(Learning song similarity metric)
  - Dataset: Million Song Dataset
  - Architecture: Similar to wavenet
FashionGAN

- A generative model, which outputs a compatible image given an input image
- Condition on the input image
- Related Work:
  - Image-to-Image Translation with Conditional Adversarial Networks
FashionGAN
Image to Image Translation with CGANs

Positive examples
Real or fake pair?

D

G tries to synthesize fake images that fool D

D tries to identify the fakes

Negative examples
Real or fake pair?

D

G
To do:

- Use Siamese encoder in Fashion GAN
- Evaluation using some subjective method
TransNets: Using Review Texts for Recommendations

- Dhruv Sharma
- Akanksha Grover
- Rishab Gulati
TransNets: Learning to Transform for Recommendation (Catherine and Cohen, 2017)
Salient Features of the Paper

❖ Learns a latent representation for the review text to predict ratings
❖ Represents a user and item as a concatenation of all reviews given by/to them
❖ Uses a CNN for Text Processing
❖ Uses Adversarial-like training technique between a source and target network
❖ Optimizes the loss over training epochs to predict accurate ratings
Dataset and Code

❖ We are using the Yelp dataset (https://www.yelp.com/dataset)

❖ Below are the statistics:
   ➢ 4,700,000 reviews
   ➢ 156,000 businesses
   ➢ 1,100,000 users

❖ For the purpose of training and testing our modifications, we will filter the users by city so that we can run the model on small portions of the dataset

❖ We are in the process of doing a proof of concept using a subset of data and in the end we will run our model on the entire dataset

❖ Code Repository for the current TransNet implementation: https://github.com/rosecatherinek/TransNets
Proposed Extensions (1)

1. **Issue**: Does not take into account the variations in reviews for the user over different items
   **Solution**: Modifying the Input Format of User/Item Review Texts embedding matrix to the CNN

   Each column is a the latent representation of a user review/item review

   *(we do this by summing the latent vectors of all words in the review)*

   - The current model concatenates all user reviews/item reviews and does take into account the variations over reviews.
Proposed Extensions (2)

2. **Issue:** Solution 1 does not take into account the interaction between different words in a sentence  
**Solution:** Make a 3D input of $|\text{Sentence Length}| \times |\text{Items}| \times |\text{Size of word Embedding}|$

- Represents each word in the review of an item by a user
- Each column is the latent representation of a user review
- Different reviews of the user/item
Proposed Extensions (3)

Using the TransNets model to generate review summary

- Inspired by the paper: “Extracting and Ranking Travel Tips from User-Generated Reviews, Guy, A Mejer, A Nus, F Raiber”
- We propose to train a RNN to produce summaries of review text given a <user,item>
- The latent representations for a review by <user,item> learnt by TransNets will be fed into the RNN
Neural Collaborative Filtering

Project Proposal CSE 291-B
Sai Kolasani, Kulshreshth Dhiman
Introduction

Figure 3: Neural matrix factorization model
Different Dataset

- The neural collaborative filtering paper uses the MovieLens & Pinterest datasets
- We plan to use the amazon reviews dataset which is sparser than movielens and more prone to cold start
- We plan to use item metadata to address the cold start issues.
Address coldstart

- We propose to try to address the cold start problem by using features from item metadata.
- The feature encoding of the items can be introduced into the NCF network and after training, the network should be able to model the preferences of users for certain item features and his should tackle the cold start problem.
- We will compare the results with NCF approach to compare the results.
Combining GMF and MLP

- NCF uses parameter ‘alpha’ which weights h_GMF and h_MLP.

\[ h \leftarrow \begin{bmatrix} \alpha h_{GMF} \\ (1 - \alpha) h_{MLP} \end{bmatrix} \]

- We propose to pre-train GMF and MLP separately instead of setting ‘alpha’ to 0.5

- We propose to weight different hidden dimensions with different weights.

- We propose to modify the network so that these weights can be learned naturally during the deep network training process rather than performing an exhaustive search for a value that works better
Experiment with different network architecture

- We would experiment with model architecture like
  - Adding hidden layer to GMF model
  - Merging the two models early to capture more interactions
Compare with Neural Factorization Machines

- Compare the this model with Neural Factorization Machines [He, Chua 2017]
  - NFM was basically non-linear Factorization Machines for rating prediction task with \{userid, itemid, context\} as a feature vector
Questions?
Jointly Modeling Aspects, Ratings and Sentiments for Movie Recommendation (JMARS)

Presented By: Rishabh Misra, Tushar Bansal
Problem

- **Motivation**: Uncovering aspects and sentiments from reviews could provide a better understanding of users, movies (items), and the process involved in generating ratings.

- **Approach**: Capture the interest distribution of users and the content distribution for movies and provide a link between interest and relevance on a per-aspect basis. Authors also differentiate between positive and negative sentiments on a per-aspect basis. This all leads to better rating prediction.
Model
Example Review

I enjoyed this DVD from the library very much. Daniel Craig plays a believable James Bond. There are some of the older 007 action scenes and similar gimmicks with updates thanks to the younger Quartermaster. Eve plays well with grit and feminism including a surprise revelation at the end. It’s touching as well with the final scenes in the mansion and the old Caretaker. Adele’s award for best song is well deserved. But the plot was pretty weak and the film dragged on and on and on, probably being 30 minutes too long. The filming is it’s usual high quality, but still overall both my wife and I found this boring, something you can’t usually level against a Bond film.

Positive sentiments are annotated as green, negative ones as red, and blue terms are movie-specific.
Algorithm

- **Objective:**
  \[ \mathcal{L}' = \sum_{r_{um} \in \mathcal{R}} \left[ \epsilon^{-2} (r_{um} - \hat{r}_{um})^2 - \log p(\{w, y, z, s\}_{um} | \Theta) \right] - \log p(\Theta | \Upsilon). \]

- **EM Algorithm**
  - **E-Step:** Sample \(\{y, z, s\}\) for each word from the current distribution
  - **M-Step:**
    - Fix sampled \(\{y, z, s\}\) for each word
    - Optimize other parameters using L-BFGS.
Extensions

● Temporal Dynamics
  ○ Idea borrowed from Collaborative Filtering with Temporal Dynamics (Koren, 2009)
  ○ Modeling temporal dynamics of user latent factors/aspect distribution with a factor of form $\alpha_u \cdot \text{sign}(t-t_u) |t-t_u|^{\beta}$

● Hierarchical Models
  ○ Adding hierarchy to language models to capture the hierarchical nature of movie topics.
  ○ Example: For a movie, an aspect violence could have sub-aspects as murders, crime, mystery etc.
Discussion
\[ \hat{r}_{um} = \mathbf{E}_{a|\theta_u, \theta_m} \left[ v_u^\top M_a v_m + b_u + b_m + b_0 \right] \]
\[ = v_u^\top \sum_a p(a|\theta_u, \theta_m) M_a \cdot v_m + b_0 + b_u + b_m \]

\[ \mathcal{L}_u^m = \varepsilon^{-2}(r_{u,m} - \hat{r}_{u,m})^2 \]
\[ - \log p(\{w, z, s\}_{um} | \theta_u, v_u, b_u, \theta_m, v_m, b_m, M_a) \]
\[ = \varepsilon^{-2}(r_{u,m} - \hat{r}_{u,m})^2 - \sum_s N_{u,m,s}^y \log p(s | \hat{r}_{um}) \]
\[ - \sum_a \sum_s N_{u,m,a,s}^y \log p(s | r_{uma}) - \sum_a N_{u,m,a}^y \log \theta_{uma}. \]
Online Factorization-based Task Recommendation with Explicit Observations

Chester Holtz
Motivation

- Crowdsourcing systems are gaining in popularity - but both workers and requesters often have difficulty finding and assigning tasks optimally such that:
  - The task is easy or worth the payment for the worker
  - The requester receives results with high quality and low noise for minimal budget.
- We can make some assumptions to model tasks and workers
  - Workers may have a hidden task preference that we want to discover
    - They may be better at doing certain tasks compared to others in a task-heterogeneous environment.
  - The worker-task matrix may have some low-rank properties
Related Work

• (Wang et al., 2017) Studied online matrix factorization with an inter-user dependency model via UCB for item recommendation.
• (Kawale et al., 2015) Performs online low-rank matrix completion, where the explore/exploit balance is achieved via Thompson sampling.
• (Zhang et al., 2015) Proposed a contextual bandit formulation to learn worker reliability for budget-constrained task assignment and recommendation in heterogeneous crowdsourcing.
• (Yuen et al., 2012) Applied online probabilistic matrix factorization for the task recommendation problem.
Proposal

• We plan to study the online heterogenous task assignment problem and exploit both implicit worker/task feedback and explicit worker/task features under budget constraints.
  • Factorization machines can leverage explicit features and feature interactions to model reconstruction. (Rendle et al., 2010)
  • Bandit-based algorithms have proven to be effective for adaptive assignment under budget constraints. (Zhang et al., 2015)
  • We can apply these algorithms to take advantage of implicit task/worker data and explicit features to iteratively complete a worker-task matrix and learn the underlying task preferences of workers.
Theoretical Analysis

- Bandit-based algorithms are typically quantified via regret defined as the expected difference between the optimal reward obtained by the oracle item selection strategy and the reward received following the algorithm.
- We hope to leverage our problem assumptions and integrate recent advances in factorization techniques for convex recovery objective.
Data and Evaluation

• Data
  • Synthetic (Wang et al., 2017)
  • Benchmark
    • UCI
    • Movielens
    • Etc.

• Evaluation
  • Accuracy / Budget

• Baseline & Comparison Algorithms
  • Naive: randomly select a task-worker pair, use majority voting.
  • BBTA
  • Online PMF
  • etc.
Worker Models in Heterogeneous Context

• **Spammer-Hammer Model**
  • A hammer gives true labels, while a spammer gives random labels. For the heterogeneous setting, each worker is a hammer on one subset of tasks but a spammer on others.

• **One-Coin Model**
  • Each worker gives true labels with a given probability - depending on task type.

• **One-Coin Model (Malicious)**
  • This model is based on the previous one, except that we add malicious label assignment: each worker is good at one subset of tasks, bad at another one, and normal at the rest.
Transnets++

Learning to Translate Better by Accounting for Higher Order Interactions
Motivation

Neural networks are predominantly used for preprocessing of data in recommender systems.

Neural factorization machines have not been evaluated in settings where the features are neurally extracted.

Goal

What effect does the inclusion of higher order interactions have on a complex feature extraction mechanism such as TransNets?
TransNets
Factorization Machines

Neural Factorization Machine

Plain Old Factorization Machine

\[ \hat{y}(x) := w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} (v_i, v_j) x_i x_j \]
Done so far

1. Dataset: Yelp Dataset 2017
   a. 4.7 million reviews
   b. TransNets paper uses only 4.1 million reviews: Filtering criteria is unclear

2. Code: [www.github.com/rosecatherinek/TransNets](http://www.github.com/rosecatherinek/TransNets)
   a. Very research oriented code
   b. Needs lot of modifications

3. Prepared the data:
   a. Reviews are concatenated for businesses and users before training the model to save GPU time
   b. Takes around 4 hours to prepare training data to run a 40 minute epoch
To do

1. Re-evaluate TransNets on Yelp dataset
2. TransNets - FM + NFM = New Model
3. Evaluate New Model on Yelp dataset
   a. We expect around 7% improvement
   b. RMSE
4. Confirm improvement from NFM using another dataset:
   a. Google Local
   b. Amazon Reviews

Bonus:

5. Implement NFM on other models that use FM to understand where higher order interactions play an important role
Questions?
Efficient Bayesian Methods
for Graph-based
Recommendation

Ajitesh Gupta, Aditi Mavalankar and Stephanie Chen
Users and Items as Bipartite Graphs

Users

U1

U2

U3

U4

Items

I1

I2

I3
3 Step paths for ranking potential items

Target user

Potential item to be recommended
Ranking items

Define ranking function $f_u$ for each user for each item within its 3 step path neighbourhood, with the help of scoring function $s$

```
input : $G = (U \cup I, E)$, $u \in U$, scoring function $s$
output: $f_u$
1 for $v \in \delta(u)$ do
2   for $w \in \delta(v)$ do
3     for $x \in \delta(w)$ do
4       $f_u(x) := f_u(x) + s(\langle u, v, w, x \rangle)$
5     end
6   end
7 end
```

**Algorithm 1:** $f_u$ computation for target user $u$. 
Reliability Prior

- Given $j \in I$, let $Y_j$ be a binary random variable that assumes 1 if $j$ receives a positive assessment and 0 otherwise, where $P(Y_j = 1) = \theta_j$.
- $R_j =$ Set of ratings of item $j$
- Intuitively, $\theta_j$ represents the unknown reliability of item $j$ within the range $(0,1)$. As $|R_j|$ increases, the Beta distribution shape tends to concentrate around its mean, then such notion of reliability turns out to be more precise.

$$P(Y_j = 1) = \theta_j$$

$$\theta_j | R_j \sim \text{Beta}(a, b)$$

where $a = \hat{a} + |R_j^+|$ and $b = \hat{b} + |R_j^-|$.

$$f(x; \alpha, \beta) = \frac{1}{B(\alpha, \beta)} x^{\alpha-1} (1 - x)^{\beta-1}$$

$x \in [0,1]$
Scoring function

- **Posterior Inequality Scoring** - Probability of the reliability of candidate item $x$ being greater than the reliability of item $v$ in the user history.
- **Posterior Prediction Scoring** - Probability of both $v$ and $x$ receiving positive assessments where we assume that $Y_v$ and $Y_x$ are independent.
- **Posterior Odds Ratio Scoring** - How large the odds of $x$ receiving a positive assessment is when compared to the odds of $v$ receiving a positive assessment.
Datasets

- BookCrossing
- MovieLens 1M
- Amazon (Cds & Vinyl, Electronics, Kindle)
- FilmTrust
- Epinions
Extensions

1. Effect of varying path lengths
2. Conditioning scoring functions on users
3. Multiple ratings
Introduction

- Deep learning based framework which can simultaneously predict precise ratings and generate tips
- For Amazon 5 core dataset (Books, Electronics and Movies & TV)
- Gated recurrent neural networks are employed to “translate” user and item latent representations into a concise sentence
  - Multi-layer perceptron network
  - Multi-task learning approach
  - Beam search algorithm
Architecture

\[(r - \hat{r})^2\]

\[-\sum_{w \in S} \log p(w)\]

Tips:
- Really
- good
- pizza
- !
- <eos>

User \(U\)
Item \(V\)

Rating Regression
Abstractive Tips Generation
Extension 1

Do the following categories have any effect on ratings?
   1. Also viewed
   2. Also bought
   3. Bought together

If so how can we include them?

Extension :
   1. Modelling them as graphs. Latent Representations of nodes in a graph.

References :
   1. node2vec: Scalable feature learning for networks Grover et al., 2016
   2. Do "Also-Viewed" Products Help User Rating Prediction? Park et al., 2017
Extension 2

Do the images have an effect on the ratings?

Do the factoid answers affect the ratings? [Electronics, Clothing]
  E.g. Answer says Yes! feature is available, but on experience we find out that it isn’t! Does it have an effect on the rating / review / tip?

Extensions:

1. Word Embeddings of the text! - Separate out the Yes and No answers
2. Pretrained representations of the images
Extension 3 - [Bonus!]

How important is time as a factor?

Extension:

1. Capturing User and Item state

References:

1. Recurrent recommender networks Wu et al. 2017
Suggestions!
Extension to Neural Collaborative Filtering

Wen Liang, Zeng Fan
Presented the NCF (Neural Network based Collaborative Filtering) Model and GMF (General Matrix Factorization) model.
Goals

1. Tackle the sparsity issue
   The original work just remove users and items with interactions less than 20

2. Consider more information
   exploit more attributes of user and items

3. Modify current model structure based on the latest study by Wang et al. (2017)
   Attributed aware deep CF model for estimating an user-item interaction
DataSet

- MovieLens
- Pinterest
- Amazon
Sparsity

- Propose sharing embedding for users or items with similar attributes.
- Try some structures to combine the sharing part and NCF part.
Consider more information from dataset

- Hashtag
- Genre
- Occupation
- Gender
- Reviews
- Etc.
- Embed those attributes and concatenate them to user/item embeddings
Model Modification

- Refer the model by Wang et al. (2017), modified from NCF model
- Attributed aware deep CF model
- Add pooling layer above embedding layer

\[
p_U = \varphi_{\text{pairwise}}(u, \{g^u_t\}) = \sum_{t=1}^{V_u} u \odot g^u_t + \sum_{t=1}^{V_u} \sum_{t'=t+1}^{V_u} g^u_t \odot g^u_{t'}
\]

\[
q_i = \varphi_{\text{pairwise}}(i, \{g^i_t\}) = \sum_{t=1}^{V_i} i \odot g^i_t + \sum_{t=1}^{V_i} \sum_{t'=t+1}^{V_i} g^i_t \odot g^i_{t'}
\]
Questions?
Extensions for Generating and Personalizing Bundle Recommendations on Steam

Yiwen Gong, Siyu Jiang and KuangHsuan Lee
Goal

1. Predict the preference rating of the item/bundle given the user
2. Recommend bundles to the given user according to their preference
3. Generate new bundles
Data

- Bundle data - existing bundle with discount info
- User-items - purchased items/bundles for each user
- User-reviews - list of reviews by users
- All-items - existing items/bundles on steam

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Users</td>
<td>88,310</td>
</tr>
<tr>
<td>Total games</td>
<td>10,978</td>
</tr>
<tr>
<td>Total game purchases</td>
<td>902,967</td>
</tr>
<tr>
<td>Total bundles</td>
<td>615</td>
</tr>
<tr>
<td>Total bundle purchases</td>
<td>87,565</td>
</tr>
<tr>
<td>Users who purchased at least one bundle</td>
<td>29,634</td>
</tr>
<tr>
<td>Games that appear in at least one bundle</td>
<td>2,819</td>
</tr>
<tr>
<td>Average bundle size</td>
<td>5.73</td>
</tr>
</tbody>
</table>
Base Method: Bayesian Personalized Ranking

- Ranking is inferred from the implicit behavior
  - Considers purchase data only
- Non-observed user-item pairs are considered negatives
- Ranks purchased item higher
BPR model - training data

1. **Item BPR**

Training data for item BPR, $D_{item}$, is a list of triplets $(u,i_p, i_n)$

- $i_p$, an item the user has purchased (positive item)
- $i_n$, an item the user hasn’t purchased (negative item).

2. **Bundle BPR**

Training data for bundle BPR, $D_{bundle}$, is a list of triplets $(u,b_p,b_n)$

- $b_p$ and $b_n$ are positive and negative bundles for the user $u$. 
BPR model - two phase training

1. Train Item BPR to get $P_u$, $Q_i$, $\beta_i$

$$\hat{x}_{u,i} = \beta_i + P_u \cdot Q_i$$

$$BPROpt(\theta) = \sum_{(u, i_p, i_n) \in D} \log(\sigma(\hat{x}_{u, i_p}(\theta) - \hat{x}_{u, i_n}(\theta))) - \lambda \| \theta \|^2$$

Maximize $BPROpt$ with gradient descent to get the parameters.
BPR model - two phase training

2. Train Bundle BPR to get parameters for $\hat{x}_{u,b}$

$$\hat{x}_{u,b} = \frac{1}{|B_b|} \sum_{i \in B_b} \left[ k\beta_i + (\mu P_u)(\omega Q_i) \right] + Cc_b + N_b$$

$$\text{BPROpt}(\theta) = \sum_{(u, i_p, i_n) \in D} \log(\sigma(\hat{x}_{u, i_p}(\theta) - \hat{x}_{u, i_n}(\theta))) - \lambda \|\theta\|^2$$

Cb represents the mean pair-wise correlation of items. Nb is used to penalize bundles with large sizes.

Maximize the BPROpt to get other parameters.
Evaluation

1. Compute the AUC to evaluate both item BPR and bundle BPR.
2. Count the ratio that the model correctly ranks $p$ higher than $n$.

\[
AUC = \frac{1}{|T|} \sum_{(u, p, n) \in T} \delta(\hat{x}_{u, i_p} - \hat{x}_{u, i_n} > 0)
\]
Personalized Bundle Generation with Greedy Algorithm

(1) Start with a bundle $b$ containing $S$ randomly-chosen items.
(2) Randomly select a set of $k$ items $I_\star \subset I \setminus \{i \in b\}$ and form a set of new bundles $B_\star$ by adding, deleting, and substituting items from $I_\star$.
(3) Let $b_\star$ be the most preferred bundle (by a user $u$) among all bundles in $B_\star$ and $P$ be the corresponding preference score, i.e., $b_\star = \arg \max_{b' \in B_\star}(\hat{x}_{u,b'} - \hat{x}_{u,b})$, and $P = \max(\hat{x}_{u,b'} - \hat{x}_{u,b})$.
(4) If $P > 0$, then accept $b_\star$ as the new bundle, otherwise accept $b_\star$ with diminishing probability (following an annealing schedule).
(5) Repeat steps 2 to 4 until convergence.
Issues with This Method

1. The original method only considers the latent variable of the bundle, some useful factors: reviews, category, manufacturer and visual factor.
2. The model also ignores the discount factor, some bundle even discount for 40%.

<table>
<thead>
<tr>
<th>Given user C</th>
<th>Bundle A</th>
<th>Bundle B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preference</td>
<td><img src="image" alt="DOTA 2" /></td>
<td><img src="image" alt="HERO OF THE KINGDOM" /></td>
</tr>
<tr>
<td>Discount</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Decision</td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>
Extensions

1. Add review data with word embedding with deep learning
2. Impose the visual image features with deep learning
3. Add category and manufacturer feature on top of latent factor model

\[ X_{u,i} = \beta_i + P_u(Q_i + A_i) + \theta_u(Ef_i) + \theta'f_i + \varphi_u g_i \]

\[ X_{u,b} = \frac{1}{B_b} \sum_{i \in B_b} [k\beta_b + \mu P_u(\omega Q_i + \alpha A_i) + \gamma \theta_u(Ef_i) + \gamma' \theta'f_i + \varphi_u g_i] + C_{cb} + N_b \]

- \( \theta_u(Ef_i) \) is the visual interaction between \( u \) and \( i \)
- \( \varphi_u g_i \) is the review interaction between \( u \) and \( i \)

a visual bias term \( \theta' \) whose inner product with \( f_i \) models users' overall opinion toward the visual appearance of a given item.

a review bias term \( \varphi_u \) whose inner product with \( g_i \) models users' overall opinion toward the review of a given item.
Extensions

1. Add review data with word embedding with deep learning
2. Impose the visual image features with deep learning
3. Add category and manufacturer feature on top of latent factor model
4. Apply consumer price sensitivity to recommend the bundles and improve the model

Reference:

1. The profit benefits of bundle pricing of complementary products
2. The Influence of Price Sensitivity, Bundle Discount Type and Price Level of Male Cosmetics on Quality Perception
TransRec: Smarter Translation Vectors

Rajiv Pasricha
Translation-based Recommendation, by Ruining He, Wang-Cheng Kang, and Julian McAuley

- Sequential model for recommendation
  - Embed users and items into a low-dimensional “translation space”
  - Each user travels along their personalized trajectory of item interactions
  - **Translation operation:** \( \text{prev. item} + \text{user} \approx \text{next item} \)
The TransRec Model

\[
\text{Prob}(j \mid u, i) \propto \beta_j - d(\tilde{y}_i + \tilde{T}_u, \tilde{y}_j),
\]
subject to \(\tilde{y}_i \in \Psi \subseteq \Phi, \\text{ for } i \in I\)

- Probability of next item \(j\) given user \(u\) and previous item \(i\)
- \(\beta_j\) = item bias (captures overall item popularity)
- \(d\) = distance function (e.g. L1 or L2)
- \(\gamma_i\) = previous item factors, \(\gamma_j\) = next item factors
- \(T_u\) = user translation vector
- \(\Phi, \Psi\) = transition space and subspace, restricting factors helps regularization
- Trained using Sequential BPR Loss, SGD

\[
\hat{\Theta} = \arg\max_{\Theta} \ln \prod_{u \in U} \prod_{j \in S_u} \prod_{j' \notin S_u} \text{Prob}(j > u, i, j' \mid \Theta) \text{Prob}(\Theta)
\]

\[
= \arg\max_{\Theta} \sum_{u \in U} \sum_{j \in S_u} \sum_{j' \notin S_u} \ln \sigma(\bar{p}_{u, i, j} - \bar{p}_{u, i, j'}) - \Omega(\Theta).
\]
Extensions

- **Personalized translation vector**
  - Model “typical” sequences of items that are common across users
    - **Current**: $\tilde{T}_u = \tilde{t} + \tilde{t}_u$
    - **Proposal**: $\tilde{T}_{u,i} = \tilde{t} + \tilde{t}_u + \tilde{t}_i$

- **Nonlinear translations**
  - More complex relationships between previous item and translation vector
    - **Current**: $d(\tilde{\gamma}_i + \tilde{T}_u, \tilde{\gamma}_j)$
    - **Proposal**: $d(f(\tilde{\gamma}_i, \tilde{T}_u), \tilde{\gamma}_j)$
  - More complex distance function that is learned by the model
    - **Current**: $d(\tilde{\gamma}_i + \tilde{T}_u, \tilde{\gamma}_j)$
    - **Proposal**: $f(\tilde{\gamma}_i, \tilde{T}_u, \tilde{\gamma}_j)$
  - What functions to use?
    - Feedforward neural networks
    - RNNs for sequence modeling?
    - etc.
Extensions

- Add Temporal Data
  - Incorporate the time delay between interactions
  - Interactions that are farther apart can have larger translations between them

- Add Content Data
  - Incorporate knowledge graph relationships as regularization
  - Items that are “related” to each other via a knowledge graph should be placed closer together in the translation space
Datasets and Evaluation

Datasets in Original Paper
- Amazon Datasets
  - Automotive, Electronics, Clothing, Jewelry, etc.
- Epinions reviews
- Foursquare check-ins
- Flixster movie ratings
- Google Local business ratings

Evaluation Metrics
- AUC
- Hit @ n
Questions?
Extension on Image-based Recommendations on Styles and Substitutes

Moyuan Huang, Yan Cheng
Paper

- (McAuley et al., 2015) Image-based recommendations on styles and substitutes
Introduction

- model this human sense of the relationships between objects based on their appearance
- modeling the human notion of which objects complement each other and which might be seen as acceptable alternatives.
- we develop a system that capable of recommending which clothes and accessories will go well together (and which will not), amongst a host of other applications.
Dataset

- based on the Amazon web store.
- contains over 180 million relationships between a pool of almost 6 million objects
- these relationships are a result of visiting Amazon and recording the product recommendations
relationships describe two specific notions of ‘compatibility’: substitute and complement goods.

- Substitute goods are those that can be interchanged
- Complements are those that might be purchased together
In the data set, relationship of 4 types:

1) ‘users who viewed X also viewed Y’ (65M edges);
2) ‘users who viewed X eventually bought Y’ (7.3M edges);
3) ‘users who bought X also bought Y’ (104M edges);
4) ‘users bought X and Y simultaneously’ (3.4M edges).

Categories 1 and 2 indicate (up to some noise) that two products may be substitutable, while 3 and 4 indicate that two products may be complementary.
Why choosing image?

- visual explanations might be useful for some categories
- the image is the most important feature for many categories
- cold-start problems
Implementation & Model

- $x$ feature generated from CNN(FC7) instead of raw pixel input: better semantic feature
- $d(x_i, x_j)$ parameterized distance metric that assigns lower value to related items, and higher to unrelated items: cluster similar commodities together for recommendation
- Shifted sigmoid function with parameter $c$

$$P(r_{ij} \in \mathcal{R}) = \sigma_c(-d(x_i, x_j)) = \frac{1}{1 + e^{d(x_i, x_j) - c}}.$$
Implementation & Model

- Potential distance functions
  - weighted nearest neighbor: giving different emphasize on different feature dimensions
  - not able to catch pair level features

\[ d_w(x_i, x_j) = \| w \circ (x_i - x_j) \|^2 \]
Implementation & Model

- Potential distance functions
  - mahalanobis transformation(style): correlate different dimension together
  - M: 4096 * 4096
  - Y: 4096 * K (K = 10, 100)

\[
M \sim Y Y^T \quad \Rightarrow \quad d_M(x_i, x_j) = (x_i - x_j)M(x_i - x_j)^T
\]

\[
d_Y(x_i, x_j) = (x_i - x_j)Y Y^T (x_i - x_j)^T = \| (x_i - x_j)Y \|^2_2.
\]

\[
\text{s}_i = x_i Y \quad \Rightarrow \quad d_Y(x_i, x_j) = \| \text{s}_i - s_j \|^2_2
\]
Implementation & Model

- Potential distance functions
  - One step further: personalized distance
    - $D(u)$ K x K diagonal matrix: $D_{kk}^{(u)}$ indicates the extent to which the user $u$ cares about the k-th dimension

\[
d_Y, u(x_i, x_j) = (x_i - x_j) Y D^{(u)} Y^T (x_i - x_j)^T
\]

\[
\longrightarrow d_Y, u(x_i, x_j) = \| (s_i - s_j) \circ X_u \|^2_2
\]
Implementation & Model

- Training phase
  - maximizing log likelihood
  - L-BFGS: quasi-newton for nonlinear optimization with too many parameters
  - R: related item set
  - Q: unrelated item set

\[
l(Y, c|\mathcal{R}, \mathcal{Q}) = \sum_{r_{ij} \in \mathcal{R}} \log(\sigma_c(-d_Y(x_i, x_j))) + \sum_{r_{ij} \in \mathcal{Q}} \log(1 - \sigma_c(-d_Y(x_i, x_j))).\]
Extension Proposal

- **Model level:** integrate some guidance to distinguish different correlated items
  - How close two items should be when they are related?
- **Feature level:** better to focus on a certain area
  - the image may contain some pixels acting as noise to the model
  - the model may focus on wrong attribute
  - Replace image features from FC7 with region proposal areas
- **Dataset:** Extend this model to food or cuisine substitute.
  - utilizing the Yelp 2017 dataset which contains 200,000 pictures
  - might confuse the model since the shape of dishes are often similar
  - the previous proposal may help
Personalized Ranking Metric Embedding (PRME)

Shreyas Udupa Balekudru
Background

- PRME-G proposed for Next New POI recommendation
- Incorporates sequential information, individual preference and geographical influence to improve recommendation performance on Location-Based Social Networks.
- Next POI recommendation is easier than Next New POI recommendation.
- Improves upon FPMC by not having independence assumption on latent vectors.
Summary of the algorithm

- Uses a pairwise Metric Embedding algorithm to model the sequential transition of POIs.
- Personalization achieved by using weighted combination of user preference latent space and sequential transition latent space.
- Embeds POIs into latent space and ranks based on Euclidian distance.
Dataset Used

FourSquare check-ins within Singapore

Gowalla check-ins within California and Nevada

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#User</th>
<th>#POI</th>
<th>#Check-in</th>
<th>Time range</th>
</tr>
</thead>
<tbody>
<tr>
<td>FourSquare</td>
<td>1917</td>
<td>2675</td>
<td>155365</td>
<td>08/2010-07/2011</td>
</tr>
<tr>
<td>Gowalla</td>
<td>4996</td>
<td>6871</td>
<td>245157</td>
<td>11/2009-10/2010</td>
</tr>
</tbody>
</table>
Incorporating Geographical Influence

PRME

\[ D_{u, l^c, l} = \begin{cases} D_{u, l}^P & \text{if } \Delta(l, l^c) > \tau \\ \alpha D_{u, l}^P + (1 - \alpha) D_{l^c, l}^S & \text{otherwise} \end{cases} \]

PRME-G

\[ D_{u, l^c, l}^G = \begin{cases} D_{u, l}^P & \text{if } \Delta(l, l^c) > \tau \\ \omega_{l^c, l} \cdot \left( \alpha D_{u, l}^P + (1 - \alpha) D_{l^c, l}^S \right) & \text{otherwise} \end{cases} \]

\[ \omega_{l^c, l} = \left( 1 + d_{l^c, l} \right)^{0.25} \]
Incorporating Geographical Influence

- This weight measure seems to be a hand-crafted function with no real physical significance.
- Can the geographic distance be used as is?

\[ w_{lc,l} = d_{lc,l} \]

- Can it be weighted using a hyperparameter?

\[ w_{lc,l} = \beta d_{lc,l} \]

- Can the geographic distance be encoded in the embedding?
Does PRME work for Product Recommendations?

- Amazon Book Ratings Dataset - 22,507,155 ratings
- Amazon Grocery Dataset - 1,297,156 ratings
- (user, item, rating, timestamp) tuples

- Does it make sense to recommend only unseen items?
- Is the performance of the method category dependent?
- Can the rating be treated as a feature (like geographic distance)?
Visualization

Embedding into a lower dimension provides interesting visualization opportunities.

Does latent space visualization provide additional insights regarding location / product similarity, user rating tendency, etc.?
Questions?
Collaborative Variational Autoencoder for Recommender Systems

Digvijay Karamchandani, Kriti Aggarwal, Sudhanshu Bahety
Original Paper

- Bayesian Generative model - Both content and rating are generated using latent variables
  - Ratings through graphical model
  - Content through generation network
Extensions

- Adding temporal Dynamics
- Also using user content and history for content based recommendation

Evaluation

\[
\text{recall@}M = \frac{\text{number of items that the user likes among the top } M}{\text{total number of items the user likes}}.
\]
Dataset

Original dataset:

Two data sets of users and their libraries of articles with different scales and degrees of sparsity obtained from CiteULike.

Our dataset:

Amazon recommendation dataset

MovieLens
Questions?