Final Presentation: Recommendation Systems

By: Alex, Anna, Navya, Paola

Agenda

- Brief Introduction of the Problem and Dataset
- Details of Model Implementations
- Experiment Results
- Future Works

Problem & Dataset

- Problem: Provide users an item recommendation of a movie based off of their own previous movie ratings and other users’ ratings

- Dataset: MovieLens 1M Dataset

  - Consists of 3 files — users.dat, ratings.dat, and movies.dat
  - 1 million ratings from 6000 users on 4000 movies
  - Each user has at least 20 ratings
  - Relatively dense dataset
  - Converted user ratings from 1-5 (explicit) to 0,1 (implicit, binary)
    - 0 if <4 and 1 if >=4

Model Implementations: NMF

- Uses utility matrix (user matrix x item matrix) in order to give recommendations based off of highly rated movies in users with similar profiles

n_components	ndcg@10
100	0.63335
400	0.63613

tolerance	ndcg@10
0.05	0.63335
0.0001	0.63609

alpha_W=alpha_H	ndcg@10
0.1	0.8182
0.01	0.8166
0.001	0.7674

Objective function:

$$
\begin{aligned}
L(W, H) &= 0.5 * ||X - WH||_{loss}^2 \\
&+ alpha\_W * l1\_ratio * n\_features * ||vec(W)||_1 \\
&+ alpha\_H * l1\_ratio * n\_samples * ||vec(H)||_1 \\
&+ 0.5 * alpha\_W * (1 - l1\_ratio) * n\_features * ||W||_{Fro}^2 \\
&+ 0.5 * alpha\_H * (1 - l1\_ratio) * n\_samples * ||H||_{Fro}^2
\end{aligned}
$$

- l1_ratio did not seem to affect ndcg score much

- Values >= 0.01 had better ndcg@10 scores, but shrunk utility matrix values to nearly zero and recommended the same movie to all users

- Smaller (0.001) alpha_W & alpha_H values were better with gently regularized factorization, still improved ndcg@10 scores, but also gave more reasonable movie recommendations

Model Implementations: Item2Vec

- Used SGNS to separate into positive and negative samples
- Captures relations between items and surrounding items, with each pair being a positive example
- Draws specific number of negative samples from each positive pair, calculating cosine similarity between negative samples and pair of items

Tuning embedding size:

embed_size	ndcg@10
30	0.442
60	0.450

Tuning batch size:

Batch size	ndcg@10
64	0.424
256	0.446

Tuning learning rate:

Learning rate	ndcg@10
0.001	0.384
0.01	0.446

Tuning epochs:

epochs	ndcg@10
10	0.446
20	0.446

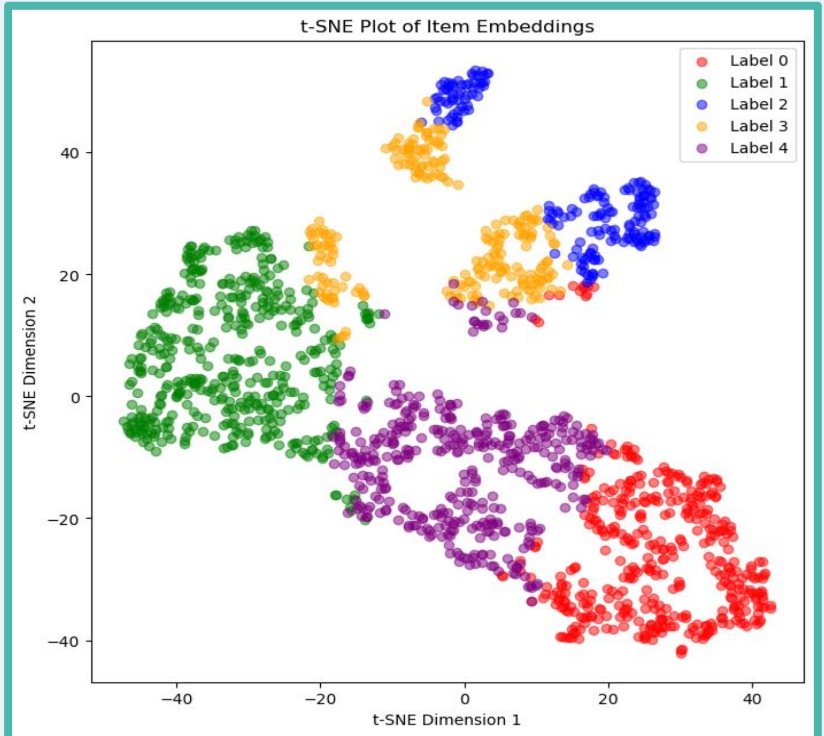
Model Implementations: t-SNE Plot

Implementation:

1. Get item embeddings from model
2. Reduce dimensionality (t-SNE)

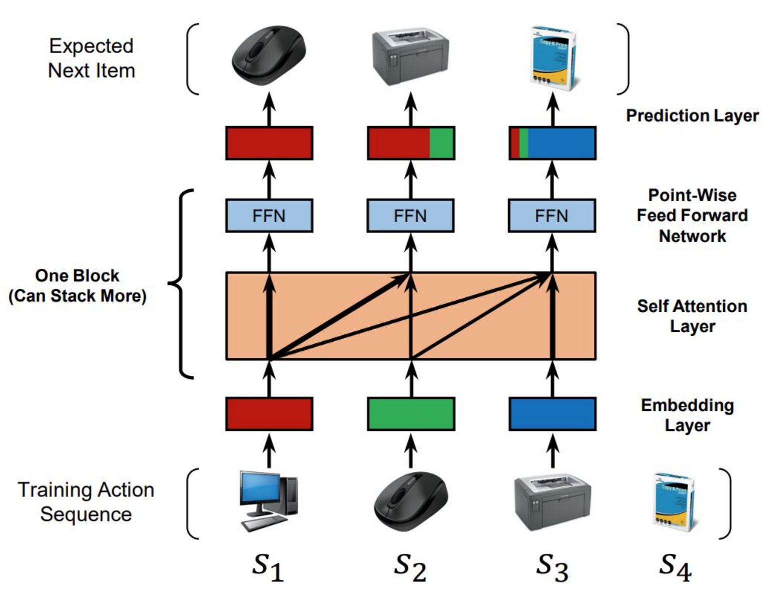
   6D → 2D/3D

3. Fit K-Means to t-SNE transformed embeddings
4. Iterate through cluster labels, find indices of embeddings, and plot



Model Implementations: SASRec

- Transformer - Uses “self-attention” instead of convolutional or recurrent algorithms
- Able to draw context from all actions in the past and frame predictions in terms of just a small number of actions
- Accomplished by adaptively assigning weights to previous items/actions at each time step
- Works well with datasets that have varying density



Kang, Wang-Cheng, and Julian McAuley. "Self-attentive sequential recommendation." 2018 IEEE international conference on data mining (ICDM). IEEE, 2018

https://github.com/pmixer/SASRec.pytorch

Model Implementations: SASRec

Default model

Component	Size
Batch size	128
Epochs	201
Dropout rate	0.2
Blocks	2

Ablation table

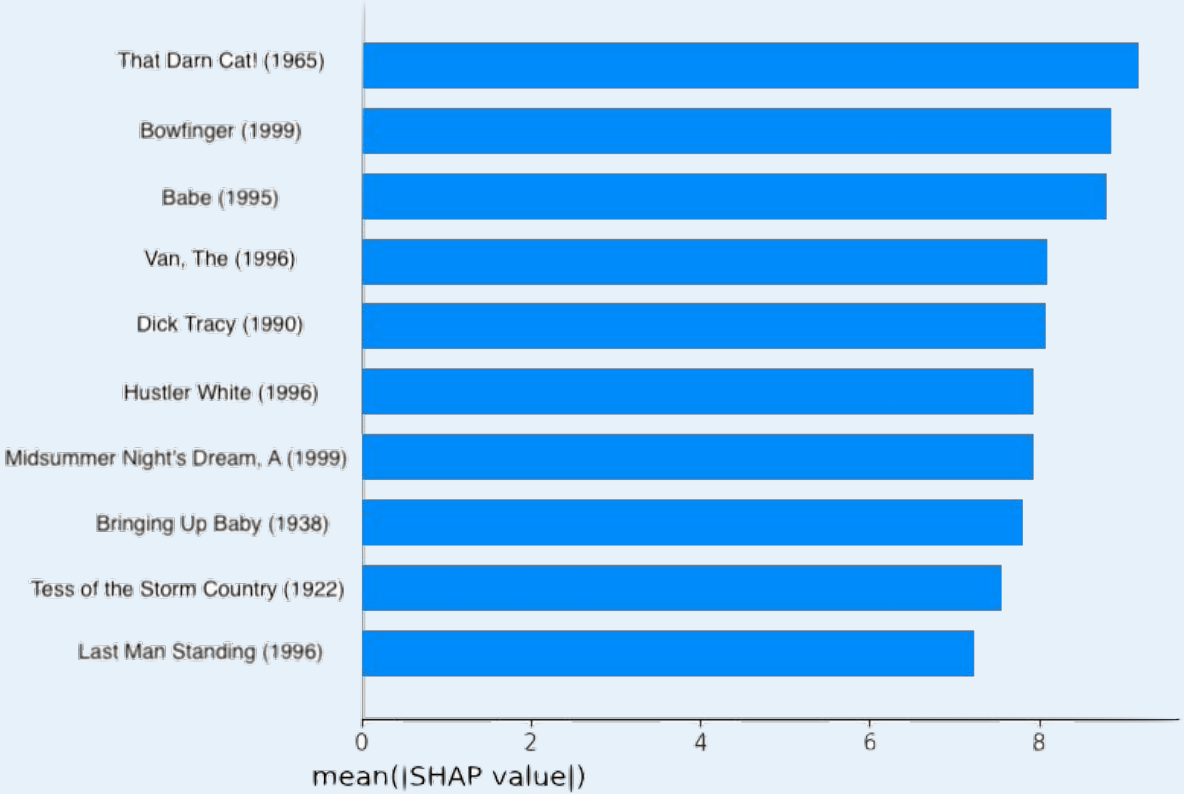
Architecture	NDCG@10
Default	0.5886
No Dropout	0.5468
0 block	0.4821
3 block	**0.5912**
Less epochs (101)	0.5834
Smaller batch size (64)	0.5878

Explaining Models

Shapley value: the average marginal contribution of a movie to the prediction across all combinations

Top Predicted Movie:
Two Girls and a Guy (1997)

Genres: the movies with the highest Shapley values tend to include one of the same genres of the predicted movie (comedy). 31% of all movies have that genre, but 70% in the top 10 do.



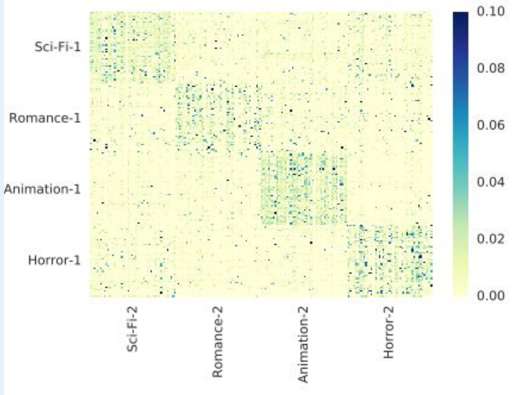
Comparing Models

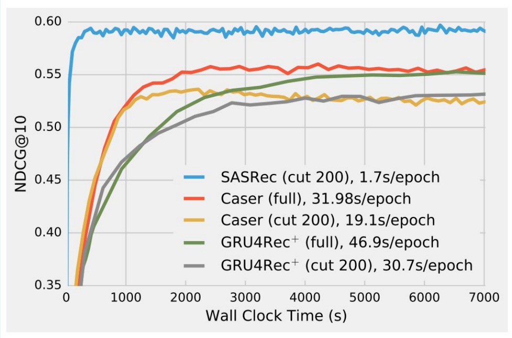
Best model: NMF
Worst model: Item2Vec
In the middle: SASRec

Model	Best NDCG@10
NMF	0.7674
Item2Vec	0.45
SASRec	0.5912

Future Work

- Evaluating each model in similar ways to the SASRec
  - Evaluate if the results share characteristics and if larger weights are assigned between similar items
  - Evaluate efficiency (time, performance) vs length of sequence
- Use large language models and prompts for movie recommendations
- Test “new” users with no/minimal data
  - Incorporating user characteristics from user.dat, such as gender, age, occupation
  - PopRec
- Using movie genre/category as a larger player in recommendation
- Evaluating other data sets





Questions?