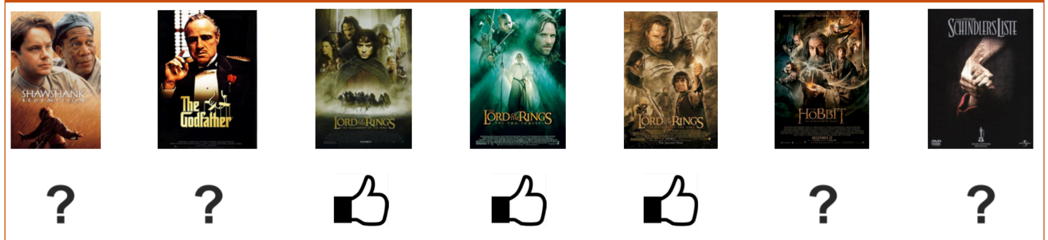
Item Recommender System Based on Collaborative Filtering

University of Florida

Team: WOCA Rong Fan, Zhaoyang Chen

Background: Recommendation

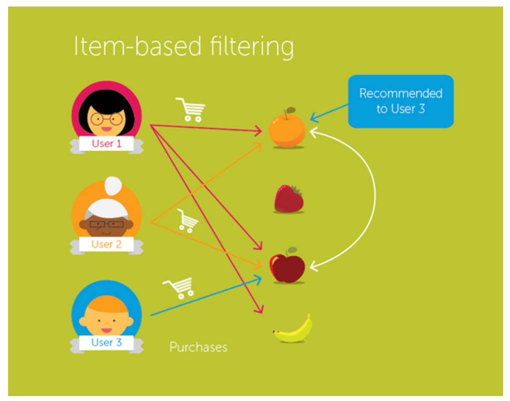


How to recommend movies to specific users based on their watched lists?

Goals

Goals: To develop a recommender system with collaborative de-noising auto-encoder neural network to provide Top-N recommendations for a list of users.

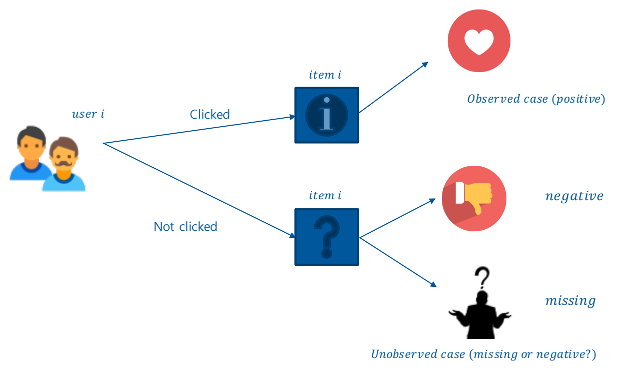
Collaborative Filtering



In general sense, collaborative filtering is the process of filtering for information or patterns using techniques involving collaboration among multiple agents, viewpoints, data sources, etc.

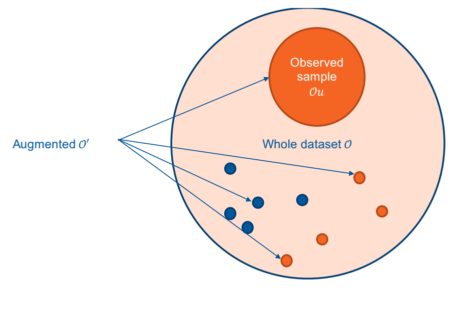
One-click Problem

In this research, we assume that one-click represents for "likes".



Negative Sampling

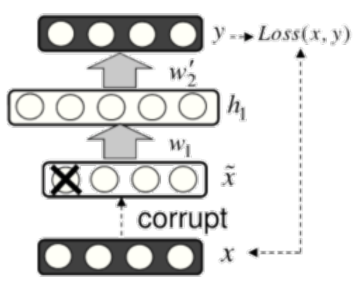
With negative sampling, we are instead going to randomly select just a small number of "negative" words (let's say 5) to update the weights for. (In this context, a "negative" word is one for which we want the network to output a 0 for).



Datasets

	#users	#items	#dyads	density(%)
MovieLens (ML)	69K	8.8K	5M	0.82
Netflix	37K	11K	4.8M	1.18
Yelp	9.6K	7K	243K	0.36

Data Corruption

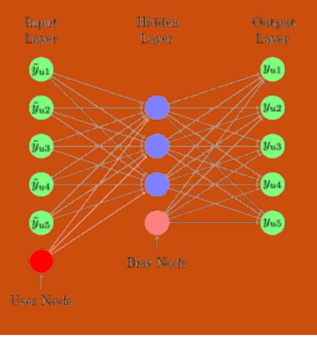


1. The corrupted input x is typically drawn from a conditional distribution.

2. Common corruption choices are the additive Gaussian noise and the multiplicative mask-out/ drop-out noise.

3. Under mask-out/drop-out corruption, one randomly overwrites each of the dimensions of x with 0 with a probability of q.

CDAE Architecture



1. In input layer., there are I+1 nodes, each of the first I nodes correspond to an item.
2. K dimensions in the hidden layer.
3. In the output layer, there are I nodes representing reconstructions of the input vector y_u.

Model Details

We load the dataset as input in our code, and use these representations to extract robust information to pass into the neural network

Step 1. Latent Factorized Model

$$\widehat{y_{ui}} = F_v(u,i) = \boldsymbol{v}_u^T \boldsymbol{v}_i$$

Step 2. Similarity Model

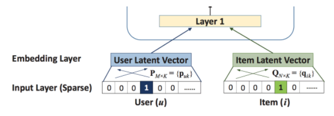
$$\widehat{y_{ui}} = F_s(u,i) = \sum_{j \in \mathcal{O}_u \setminus \{i\}} \boldsymbol{y}_{uj} \boldsymbol{S}_{ji}$$

Step 3. Factorized Similarity Model

$$\widehat{y_{ui}} = F_{p,q}(u,i) = \left(\sum\nolimits_{j \in \mathcal{O}_u \setminus \{i\}} \boldsymbol{y}_{uj} \boldsymbol{p}_j\right)^T \boldsymbol{q}_i$$

Final Step. Latent Factorized Similarity Model

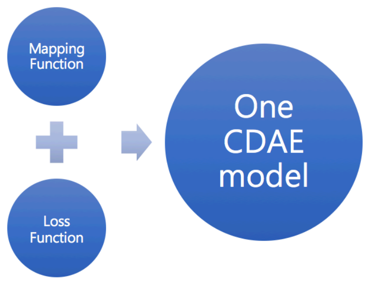
$$\widehat{y_{ui}} = F_{p,q}(u,i) = \left(\sum_{j \in \mathcal{O}_u \setminus \{i\}} \boldsymbol{y}_{uj} \boldsymbol{p}_j + \boldsymbol{p}_u\right)^T \boldsymbol{q}_i$$



Model Combinations

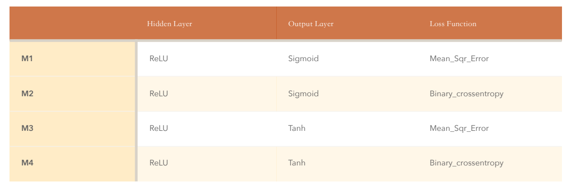
- Identity function (Input/ Output)
- Sigmoid function (Input/ Output)

- Logistic loss function
- Square loss function
- Cross Entropy loss function

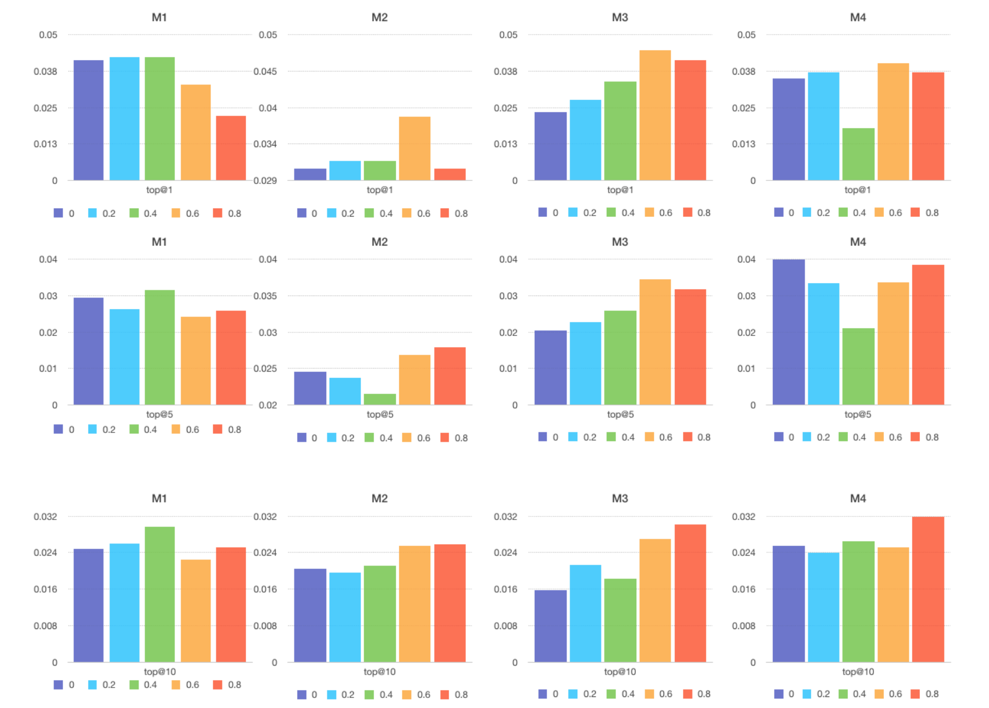


Experiment Results

Following are the experiment results for four possible variants of the CDAE model.



	Hidden Layer	Output Layer	Loss Function
M1	ReLU	Sigmoid	Mean_Sqr_Error
M2	ReLU	Sigmoid	Binary_crossentropy
M3	ReLU	Tanh	Mean_Sqr_Error
M4	ReLU	Tanh	Binary_crossentropy



Conclusion

1. For Top@1 recommendation, model 1, which is the combination of ReLU at hidden layer and sigmoid at output layer, and the mean squared error loss function has clearly outperformed others.

2. For Top@1 and Top@5 recommendations, the combination of ReLU at hidden layer and sigmoid at output layer, and the mean squared error loss function has the most satisfying results; as for the Top@10 recommendation, the combination of sigmoid at hidden layer and sigmoid at output layer, and the binary cross entropy loss function is also practical. So, the choice mostly depends on the actual practice.