AutoRec: Autoencoders Meet Collaborative Filtering

Collaborative Deep Learning for Recommender Systems

2021 DSAIL Winter Internship

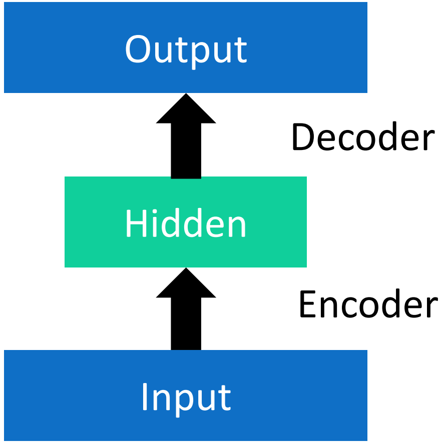
2022.02.08. Daeyoung Kim

INDEX

- Background
- Introduction
- Model
- Experiments
- Conclusions
- Further discussion

Background

Autoencoder



Unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data

Input and output have **same** dimensions

Hidden layer has **smaller** dimensions

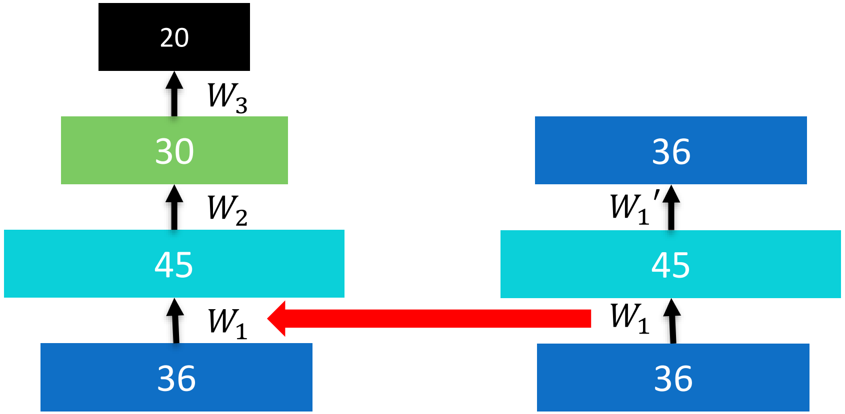
-> capture dense, meaningful factors (dimensionality reduction)

Train such that features can be used to **reconstruct** original data

Background

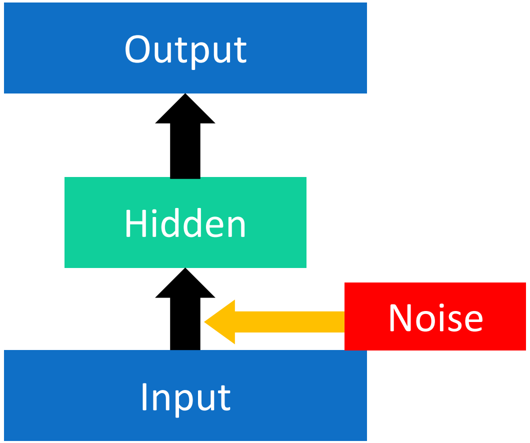
Stacked / Denoising Autoencoder

Stacked Autoencoder



- Autoencoder with multiple hidden layers
- learn weights sequentially using Autoencoder
- use for initializing weights

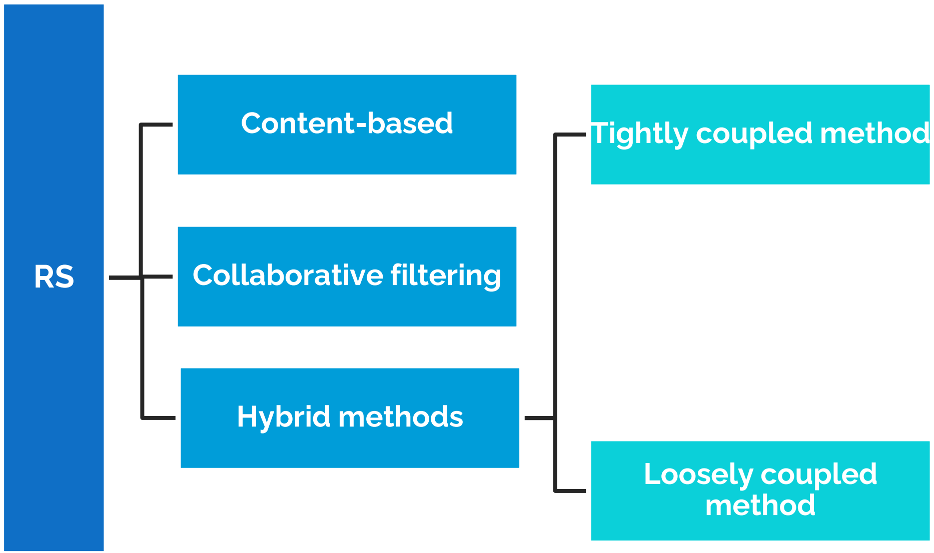
Denoising Autoencoder



- add noise to input data
- minimize error between clear data and corrupted data

Background

Recommender Systems



: allows 2-way interaction between rating, auxiliary information

-> better performance

Collaborative topic regression(**CTR**)

Introduction

Deep Learning with Collaborative Filtering

Collaborative Filtering

CTR: interaction between content and ratings

performance degrades when auxiliary information is sparse

Deep Learning models

strength: learning features automatically

weakness: learning similarity and implicit relationship between items

Integrate deep learning with CF

Introduction

About Model

AutoRec

apply Autoencoder framework to collaborative filtering method

CDL

probabilistic collaborative filtering method based on SDAE

Jointly performs representation learning and collaborative filtering

Introduction

Contributions

AutoRec

representational & computational advantages, performance improvement

CDL

Extract effective feature representations & capture similarity, implicit relationship **simultaneously**

A new probabilistic model for deep learning

1st hierarchical Bayesian model : bridge the gap between RS and state-of-the-art deep learning models

Significantly advance the state of the art

Model

AutoRec

Goal

design item-based autoencoder to predict missing ratings

Solve $\min_{\theta} \sum_{\boldsymbol{r} \in \boldsymbol{S}} ||\boldsymbol{r} - h(\boldsymbol{r};\theta)||_2^2$ where $h(\boldsymbol{r};\theta) = f(\boldsymbol{W} \cdot g(\boldsymbol{V}\boldsymbol{r} + \boldsymbol{\mu}) + \boldsymbol{b})$

Notation

m users, n ratings

$R \in \mathbb{R}^{m \times n}$: user-item rating matrix

$\boldsymbol{r}^{(u)} = (R_{u1}, \cdots, R_{un}) \in \mathbb{R}^{n}$: user vector

$\boldsymbol{r}^{(i)} = (R_{1i}, \cdots, R_{mi}) \in \mathbb{R}^{m}$: item vector

partially observed

$\boldsymbol{W} \in \mathbb{R}^{d \times k}$, $\boldsymbol{V} \in \mathbb{R}^{k \times d}$: transformations of encoder, decoder

$\boldsymbol{\mu} \in \mathbb{R}^{k}$, $\boldsymbol{b} \in \mathbb{R}^{d}$: biases of encoder, decoder

$f(\cdot), g(\cdot)$: activation functions

k : dimension of **single** hidden layer

Model

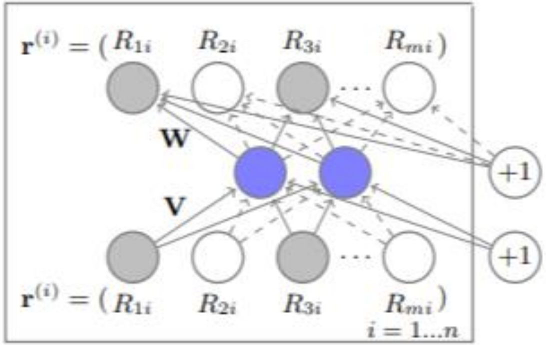
AutoRec

For all items, networks share parameters

Training

1. Update parameters only for observed inputs
2. Regularize parameters to prevent overfitting

Objective: $\min_{\theta} \sum_{i=1}^{n} ||\boldsymbol{r}^{(i)} - h(\boldsymbol{r}^{(i)};\theta)||_{O}^{2} + \frac{\lambda}{2} \cdot (||\boldsymbol{W}||_{F}^{2} + ||\boldsymbol{V}||_{F}^{2})$



I-AutoRec requires $2mk + m + k$ parameters

predicted rating for user u and item i: $\widehat{R_{ui}} = \left(h(\boldsymbol{r}^{(i)};\hat{\theta})\right)_{u}$

Model

Comparison

AutoRec	RBM-CF
discriminative model (Autoencoder based)	probabilistic model (RBM based)
minimize RMSE directly	estimate parameters by MLE
gradient-based backpropagation	contrastive divergence
Fewer parameters	More parameters
Embed users **or** items	Embed **both** users and items
Nonlinear latent representations are possible	Only linear latent representations

Model

CDL

Problem Definition

Given observed ratings and content information, predict missing ratings

Notation

I users, J items, vocabulary size of S

L: number of layers

$\boldsymbol{R} = \left[R_{ij}\right]_{I\times J}$: rating matrix

$\boldsymbol{X}_c$: content information (size: J×S), $\boldsymbol{X}_0$: noise-corrupted matrix (size: J×S)

$\boldsymbol{X}_l$: output of layer l of SDAE (size: J × K_1)

$\boldsymbol{W}_l$, $\boldsymbol{b}_l$: weight matrix, bias vector of layer l

$\boldsymbol{W}^+$: collection of all layers of weight matrices, biases

Model

Stacked Denoising Autoencoders

Given corrupted data $\boldsymbol{X}_0$, predict clear input $\boldsymbol{X}_c$

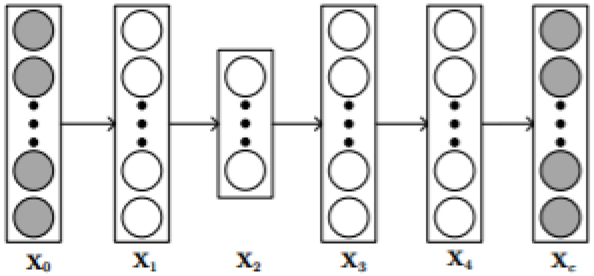


Figure 2: A 2-layer SDAE with $L = 4$.

Solve optimization problem

$$\min_{\{\mathbf{W}_l\},\{\mathbf{b}_l\}} \|\mathbf{X}_c - \mathbf{X}_L\|_F^2 + \lambda \sum_l \|\mathbf{W}_l\|_F^2$$

where λ: regularization parameter, $||\cdot||_F$: Frobenius norm

Model

Generalized Bayesian SDAE

Assume $\boldsymbol{X}_0$ and $\boldsymbol{X}_c$ are observed

Procedures

1. For each layer l of SDAE,

$$\boldsymbol{W}_{l,*n} \sim N\left(\boldsymbol{0}, {\lambda_\omega}^{-1}\boldsymbol{I}_{K_l}\right)$$

$$\boldsymbol{b}_l \sim N\left(\boldsymbol{0}, {\lambda_\omega}^{-1}\boldsymbol{I}_{K_l}\right)$$

$$\boldsymbol{X}_{l,j*} \sim N\left(\sigma\left(\boldsymbol{X}_{l-1,j*}\boldsymbol{W}_l + \boldsymbol{b}_l\right), {\lambda_s}^{-1}\boldsymbol{I}_{K_l}\right)$$

2. For each item j, $\boldsymbol{X}_{c,j*} \sim N\left(\boldsymbol{X}_{L,j*}, {\lambda_n}^{-1}\boldsymbol{I}_J\right)$

If $\lambda_s \rightarrow \infty$, model degenerates to SDAE

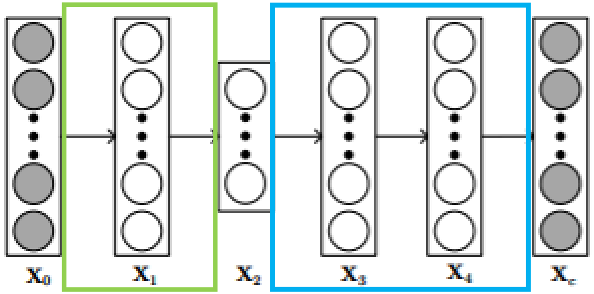


Figure 2: A 2-layer SDAE with $L = 4$.

: first $L/2$ layers act as encoder

: last $L/2$ layers act as decoder

Maximize posterior probability <->

Minimize reconstruction error with weight decay

Model

Collaborative Deep Learning

Procedures

1. For each layer l of SDAE,

$$\boldsymbol{W}_{l,*n} \sim N\left(\boldsymbol{0}, \lambda_\omega{}^{-1}\boldsymbol{I}_{K_l}\right)$$

$$\boldsymbol{b}_l \sim N\left(\boldsymbol{0}, \lambda_\omega{}^{-1}\boldsymbol{I}_{K_l}\right)$$

$$\boldsymbol{X}_{l,j*} \sim N\left(\sigma\left(\boldsymbol{X}_{l-1,j*}\boldsymbol{W}_l + \boldsymbol{b}_l\right), \lambda_s{}^{-1}\boldsymbol{I}_{K_l}\right)$$

2. For each item j, $\boldsymbol{X}_{c,j*} \sim N\left(\boldsymbol{X}_{L,j*}, \lambda_n{}^{-1}\boldsymbol{I}_J\right)$

3. Generate latent vector for **item j:**

$\boldsymbol{\epsilon}_j \sim N\left(\boldsymbol{0}, \lambda_v{}^{-1}\boldsymbol{I}_K\right)$ ($\boldsymbol{\epsilon}_j$: offset vector)

$$\boldsymbol{v}_j = \boldsymbol{\epsilon}_j + \boldsymbol{X}^T_{\frac{L}{2},j*}$$

4. Generate latent vector for **user i:**

$$\boldsymbol{u}_i \sim N\left(\boldsymbol{0}, \lambda_u{}^{-1}\boldsymbol{I}_K\right)$$

5. Generate rating for user-item pair (i, j):

$$\boldsymbol{R}_{ij} \sim N\left(\boldsymbol{u}_i{}^T\boldsymbol{v}_j, \boldsymbol{C}_{ij}{}^{-1}\right)$$

-> Using Bayesian SDAE as a component

Notation

$$\boldsymbol{C}_{ij} = \begin{cases} a, & \boldsymbol{R}_{ij} = 1 \\ b, & \boldsymbol{R}_{ij} = 0 \end{cases}$$

$\lambda_w, \lambda_n, \lambda_u, \lambda_s, \lambda_v$: hyperparameters which control **variances**

$\boldsymbol{X}_{L/_2}$: bridge between ratings and content information

For computational efficiency, take $\lambda_s \to \infty$

Model

MAP estimates

Maximize joint log-likelihood of $\boldsymbol{U}, \boldsymbol{V}, \{\boldsymbol{X}_l\}, \boldsymbol{X}_c, \boldsymbol{W}^+, \boldsymbol{R}$

$$
\begin{aligned}
\mathscr{L} = & -\frac{\lambda_u}{2}\sum_i \|\mathbf{u}_i\|_2^2 - \frac{\lambda_w}{2}\sum_l (\|\mathbf{W}_l\|_F^2 + \|\mathbf{b}_l\|_2^2) \\
& -\frac{\lambda_v}{2}\sum_j \|\mathbf{v}_j - \mathbf{X}_{\frac{L}{2},j*}^T\|_2^2 - \frac{\lambda_n}{2}\sum_j \|\mathbf{X}_{L,j*} - \mathbf{X}_{c,j*}\|_2^2 \\
& -\frac{\lambda_s}{2}\sum_l \sum_j \|\sigma(\mathbf{X}_{l-1,j*}\mathbf{W}_l + \mathbf{b}_l) - \mathbf{X}_{l,j*}\|_2^2 \\
& -\sum_{i,j} \frac{\mathbf{C}_{ij}}{2}(\mathbf{R}_{ij} - \mathbf{u}_i^T \mathbf{v}_j)^2.
\end{aligned}
$$

If $\lambda_s \to \infty$,

$$
\begin{aligned}
\mathscr{L} = & \boxed{-\frac{\lambda_u}{2}\sum_i \|\mathbf{u}_i\|_2^2 - \frac{\lambda_w}{2}\sum_l (\|\mathbf{W}_l\|_F^2 + \|\mathbf{b}_l\|_2^2)} \; 1 \\
& \boxed{-\frac{\lambda_v}{2}\sum_j \|\mathbf{v}_j - f_e(\mathbf{X}_{0,j*}, \mathbf{W}^+)^T\|_2^2} \; 2 \\
& \boxed{-\frac{\lambda_n}{2}\sum_j \|f_r(\mathbf{X}_{0,j*}, \mathbf{W}^+) - \mathbf{X}_{c,j*}\|_2^2} \; 3 \\
& \boxed{-\sum_{i,j} \frac{\mathbf{C}_{ij}}{2}(\mathbf{R}_{ij} - \mathbf{u}_i^T \mathbf{v}_j)^2,} \; 4
\end{aligned}
$$

Notation

$f_e(\cdot, \boldsymbol{W}^+)$: computes encoding of item

$f_r(\cdot, \boldsymbol{W}^+)$: f_e+ reconstructed content vector of item

Explanation

1: regularization for $\boldsymbol{u}$, $\boldsymbol{W}^+$

2: optimize $\boldsymbol{v}$

3: minimize reconstruction error

4: error of predicted ratings

Model

MAP estimates & Prediction

Update rules

$$\mathbf{u}_i \leftarrow (\mathbf{V}\mathbf{C}_i\mathbf{V}^T + \lambda_u \mathbf{I}_K)^{-1}\mathbf{V}\mathbf{C}_i\mathbf{R}_i$$

$$\mathbf{v}_j \leftarrow (\mathbf{U}\mathbf{C}_j\mathbf{U}^T + \lambda_v \mathbf{I}_K)^{-1}(\mathbf{U}\mathbf{C}_j\mathbf{R}_j + \lambda_v f_e(\mathbf{X}_{0,j*}, \mathbf{W}^+)^T)$$

Learning weights, biases using back-propagation

Gradients of likelihood

$$\nabla_{\mathbf{W}_l}\mathscr{L} = -\lambda_w \mathbf{W}_l$$
$$-\lambda_v \sum_j \nabla_{\mathbf{W}_l} f_e(\mathbf{X}_{0,j*}, \mathbf{W}^+)^T (f_e(\mathbf{X}_{0,j*}, \mathbf{W}^+)^T - \mathbf{v}_j)$$
$$-\lambda_n \sum_j \nabla_{\mathbf{W}_l} f_r(\mathbf{X}_{0,j*}, \mathbf{W}^+)(f_r(\mathbf{X}_{0,j*}, \mathbf{W}^+) - \mathbf{X}_{c,j*})$$

$$\nabla_{\mathbf{b}_l}\mathscr{L} = -\lambda_w \mathbf{b}_l$$
$$-\lambda_v \sum_j \nabla_{\mathbf{b}_l} f_e(\mathbf{X}_{0,j*}, \mathbf{W}^+)^T (f_e(\mathbf{X}_{0,j*}, \mathbf{W}^+)^T - \mathbf{v}_j)$$
$$-\lambda_n \sum_j \nabla_{\mathbf{b}_l} f_r(\mathbf{X}_{0,j*}, \mathbf{W}^+)(f_r(\mathbf{X}_{0,j*}, \mathbf{W}^+) - \mathbf{X}_{c,j*}).$$

Prediction

D: observed test data

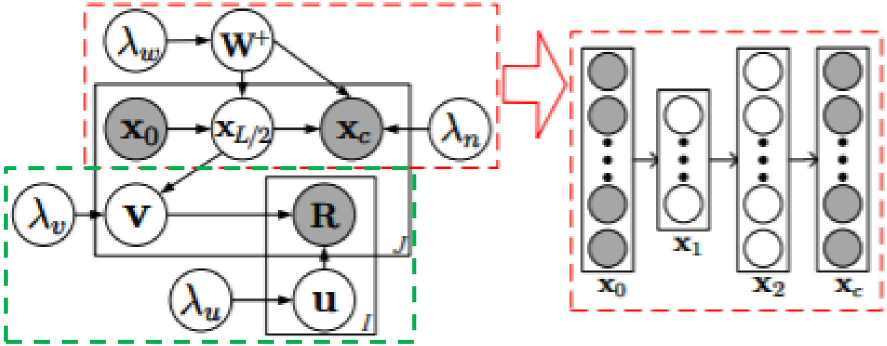
$$E[\mathbf{R}_{ij}|D] \approx E[\mathbf{u}_i|D]^T (E[f_e(\mathbf{X}_{0,j*}, \mathbf{W}^+)^T|D] + E[\boldsymbol{\epsilon}_j|D])$$

Approximation of predicted rating:

$$\mathbf{R}_{ij}^* \approx (\mathbf{u}_j^*)^T (f_e(\mathbf{X}_{0,j*}, \mathbf{W}^{+*})^T + \boldsymbol{\epsilon}_j^*) = (\mathbf{u}_i^*)^T \mathbf{v}_j^*.$$

Model

Graphical Model



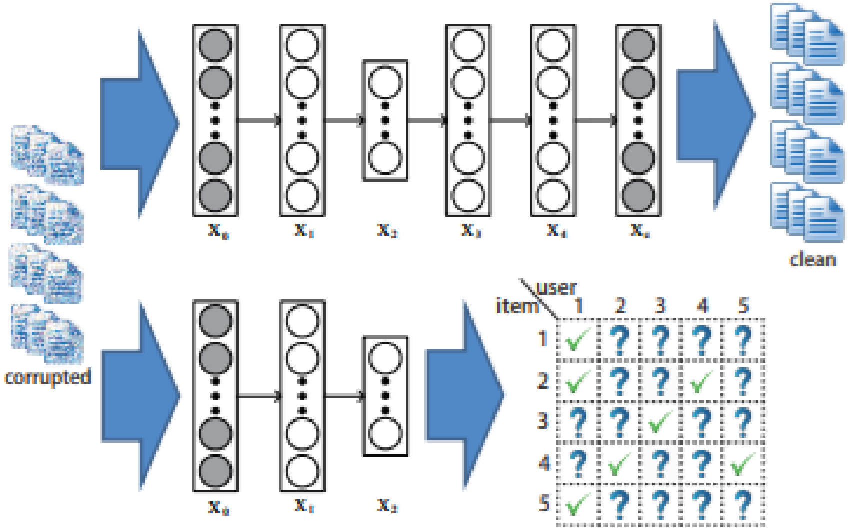
Feature learning component (SDAE)

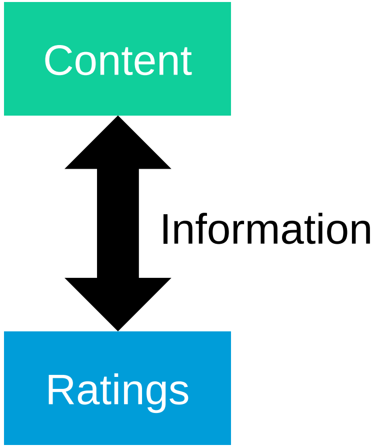
Collaborative filtering component (CTR)

Learn latent representations using **both** components

Model

Degenerated CDL





Representation learning <-> Recommendation

Experiments

AutoRec

Experiment Setup

- Baselines: RBM-CF, BiasedMF, LLORMA
- Use Movielens (1M, 10M), Netflix datasets
- default rating of 3 for test users/items without training observations
- 90%-10% train-test sets, hold out 10% of train data for validation
- Repeat 5 times, report average RMSE
- 95% C.I. for RMSE can’t exceed 0.003
- Regularization strength($\boldsymbol{\lambda}$) ranges 0.001 to 1000
- Latent dimension($\boldsymbol{k}$) ranges 10 to 500

Experiments

Results

	ML-1M	ML-10M
U-RBM	0.881	0.823
I-RBM	0.854	0.825
U-AutoRec	0.874	0.867
I-AutoRec	**0.831**	**0.782**

(a)

$f(\cdot)$	$g(\cdot)$	RMSE
Identity	Identity	0.872
Sigmoid	Identity	0.852
Identity	Sigmoid	**0.831**
Sigmoid	Sigmoid	0.836

(b)

Q1. Which is better, item- or user-based autoencoding with RBMS or AutoRec?

avg(ratings per item) > avg(ratings per user) -> **low variance** of ratings

I-AutoRec performs best

Q2. How does AutoRec performance vary with linear and nonlinear activation functions?

$$h(\boldsymbol{r};\theta) = f(\boldsymbol{W} \cdot g(\boldsymbol{V}\boldsymbol{r} + \boldsymbol{\mu}) + \boldsymbol{b})$$

Nonlinearity in the **hidden** layer ($g(\cdot)$) is critical

$f(\cdot)$: identity, $g(\cdot)$: sigmoid shows best performance

-> used for all other experiments

Experiments

Results

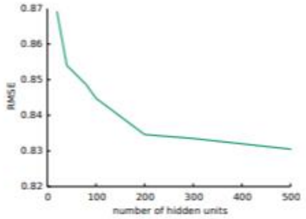


Figure 2: RMSE of I-AutoRec on Movielens 1M as the number of hidden units k varies.

	ML-1M	ML-10M	Netflix
BiasedMF	0.845	0.803	0.844
I-RBM	0.854	0.825	-
U-RBM	0.881	0.823	0.845
LLORMA	0.833	**0.782**	0.834
I-AutoRec	**0.831**	**0.782**	**0.823**

(c)

Q3. How does performance of AutoRec vary with the number of hidden unit?

Performance increases with more hidden units

Use $k = 500$ for all other experiments

Q4. How does AutoRec perform against all baselines?

AutoRec performs **best** in most experiments

Q5. Do deep extensions of AutoRec help?

Using 3 hidden layers, RMSE reduces from 0.831 to 0.827

potential for further improvement

Experiments

Datasets

CiteULike

allows users to create their own collections of articles

citeulike-t is relatively sparse than *citeulike-a*

Netflix

- movie rating dataset

Choose top S discriminative words by tf-idf values (S: 8000, 20000, 20000)

Datasets	users	items	ratings
citeulike-a	5551	16980	204987
citeulike-t	7947	25975	134860
Netflix	407261	9228	15348808

Experiments

Experiment setup

Randomly select P items associated with each user

P=1: sparse, P=10: dense

Evaluation metrics

recall@M $= \frac{\text{number of items that the user likes among the top M}}{\text{total number of items that the user likes}}$

mAP=$\frac{\sum_{q=1}^{Q} AveP(q)}{Q}$ (AveP=$\frac{\sum_{k=1}^{n} P(k) \times rel(k)}{number\ of\ relevant\ documents}$)

Baselines

CMF

SVDFeature

DeepMusic

CTR

Model Settings

Masking noise level: 0.3

Dropout rate: 0.1 (when $L > 2$)

a = 1, b = 0.01

$K_L = 200$

$K = 50$

Experiments

Quantitative Comparison

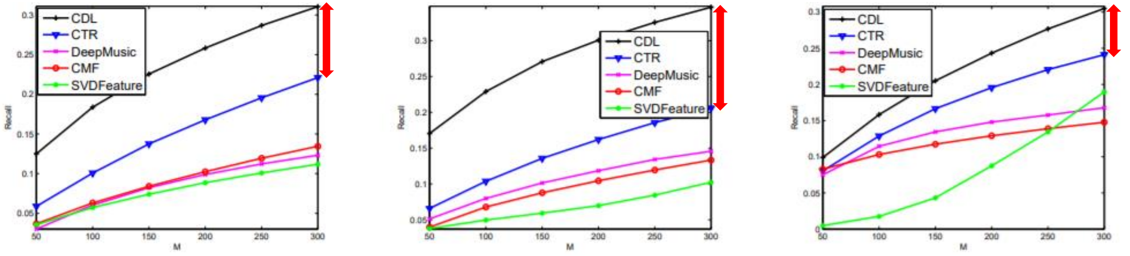


Figure 4: Performance comparison of CDL, CTR, DeepMusic, CMF, and SVDFeature based on recall@M for datasets *citeulike-a*, *citeulike-t*, and *Netflix* in the sparse setting. A 2-layer CDL is used.

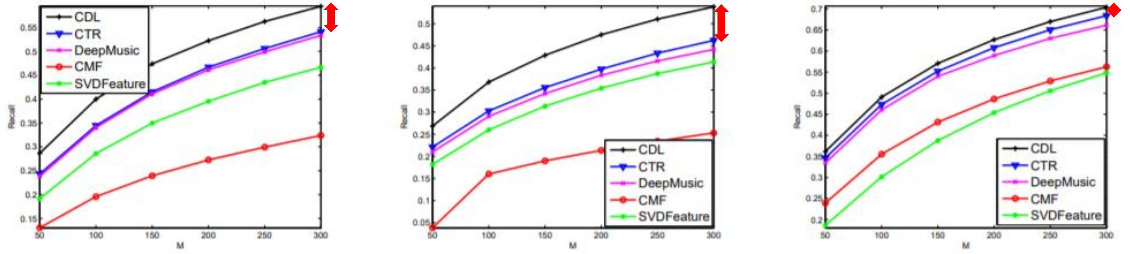


Figure 5: Performance comparison of CDL, CTR, DeepMusic, CMF, and SVDFeature based on recall@M for datasets *citeulike-a*, *citeulike-t*, and *Netflix* in the dense setting. A 2-layer CDL is used.

Comparing CDL with CTR, results show **significant** performance boost for all experiments

Experiments

Quantitative Comparison

Table 1: mAP for three datasets

	citeulike-a	*citeulike-t*	*Netflix*
CDL	**0.0514**	**0.0453**	**0.0312**
CTR	0.0236	0.0175	0.0223
DeepMusic	0.0159	0.0118	0.0167
CMF	0.0164	0.0104	0.0158
SVDFeature	0.0152	0.0103	0.0187

mAP in sparse settings

Table 2: Recall@300 in the sparse setting (%)

#layers	1	2	3
citeulike-a	27.89	**31.06**	30.70
citeulike-t	32.58	34.67	**35.48**
Netflix	29.20	30.50	**31.01**

Table 3: Recall@300 in the dense setting (%)

#layers	1	2	3
citeulike-a	58.35	**59.43**	59.31
citeulike-t	52.68	53.81	**54.48**
Netflix	69.26	70.40	**70.42**

CDL shows best performance, especially effective for sparse datasets

Deeper model performs better (beware of overfitting issues)

Experiments

Quantitative Comparison

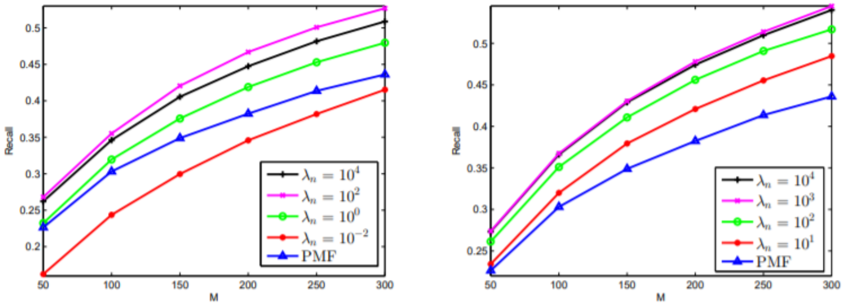


Figure 6: Performance of CDL based on recall@M for different values of λ_n on *citeulike-t*. The left plot is for $L = 2$ and the right one is for $L = 6$.

Extreme values of λ_n degrade performance

λ_n is extremely large : model degenerates to SDAE, CTR -> no interaction

λ_n is extremely small : decoder vanishes -> encoder easily overfits the latent item vectors

Experiments

Qualitative Comparison

Table 4: Interpretability of the latent structures learned

	user I (CDL)	in user's lib?
top 3 topics	1. search, image, query, images, queries, tagging, index, tags, searching, tag	
	2. social, online, internet, communities, sharing, networking, facebook, friends, ties, participation	
	3. collaborative, optimization, filtering, recommendation, contextual, planning, items, preferences	
top 10 articles	**1. The structure of collaborative tagging Systems**	**yes**
	2. Usage patterns of collaborative tagging systems	**yes**
	3. Folksonomy as a complex network	no
	4. HT06, tagging paper, taxonomy, Flickr, academic article, to read	**yes**
	5. Why do tagging systems work	**yes**
	6. Information retrieval in folksonomies: search and ranking	no
	7. tagging, communities, vocabulary, evolution	**yes**
	8. The complex dynamics of collaborative tagging	**yes**
	9. Improved annotation of the blogosphere via autotagging and hierarchical clustering	no
	10. Collaborative tagging as a tripartite network	**yes**
	user I (CTR)	in user's lib?
top 3 topics	1. social, online, internet, communities, sharing, networking, facebook, friends, ties, participation	
	2. search, image, query, images, queries, tagging, index, tags, searching, tag	
	3. feedback, event, transformation, wikipedia, indicators, vitamin, log, indirect, taxonomy	
top 10 articles	**1. HT06, tagging paper, taxonomy, Flickr, academic article, to read**	**yes**
	2. Structure and evolution of online social networks	no
	3. Group formation in large social networks: membership, growth, and evolution	no
	4. Measurement and analysis of online social networks	no
	5. A face(book) in the crowd: social searching vs. social browsing	no
	6. The strength of weak ties	no
	7. Flickr tag recommendation based on collective knowledge	no
	8. The computer-mediated communication network	no
	9. Social capital, self-esteem, and use of online social network sites: A longitudinal analysis	no
	10. Increasing participation in online communities: A framework for human-computer interaction	no
	user II (CDL)	in user's lib?
top 3 topics	1. flow, cloud, codes, matter, boundary, lattice, particles, galaxies, fluid, galaxy	
	2. mobile, membrane, wireless, sensor, mobility, lipid, traffic, infrastructure, monitoring, ad	
	3. hybrid, orientation, stress, fluctuations, load, temperature, centrality, mechanical, two-dimensional, heat	
top 10 articles	**1. Modeling the flow of dense suspensions of deformable particles in three dimensions**	**yes**
	2. Simplified particulate model for coarse-grained hemodynamics simulations	**yes**
	3. Lattice Boltzmann simulations of blood flow: non-newtonian rheology and clotting processes	**yes**
	4. A genome-wide association study for celiac disease identifies risk variants	**yes**
	5. Efficient and accurate simulations of deformable particles	**yes**
	6. A multiscale model of thrombus development	**yes**
	7. Multiphase hemodynamic simulation of pulsatile flow in a coronary artery	**yes**
	8. Lattice Boltzmann modeling of thrombosis in giant aneurysms	**yes**
	9. A lattice Boltzmann simulation of clotting in stented aneursysms	**yes**
	10. Predicting dynamics and rheology of blood flow	**yes**
	user II (CTR)	in user's lib?
top 3 topics	1. flow, cloud, codes, matter, boundary, lattice, particles, galaxies, fluid, galaxy	
	2. transition, equations, dynamical, discrete, equation, dimensions, chaos, transitions, living, trust	
	3. mobile, membrane, wireless, sensor, mobility, lipid, traffic, infrastructure, monitoring, ad	
top 10 articles	**1. Multiphase hemodynamic simulation of pulsatile flow in a coronary artery**	**yes**
	2. The metallicity evolution of star-forming galaxies from redshift 0 to 3	no
	3. Formation versus destruction: the evolution of the star cluster population in galaxy mergers	no
	4. Clearing the gas from globular clusters	no
	5. Macroscopic effects of the spectral structure in turbulent flows	no
	6. The WiggleZ dark energy survey	no
	7. Lattice-Boltzmann simulation of blood flow in digitized vessel networks	no
	8. Global properties of 'ordinary' early-type galaxies	no
	9. Proteus : a direct forcing method in the simulations of particulate flows	**yes**
	10. Analysis of mechanisms for platelet near-wall excess under arterial blood flow conditions	**yes**

Precision Comparison (P=1, sparse)

	CDL	CTR
User I	70%	10%
User II	100%	30%

CDL captures the key points of articles, user preferences more accurately (User I)

CDL can model the co-occurrence and relations of words better (User II)

Experiments

Qualitative Comparison

Table 5: Example user with recommended movies

User III	Movies in the training set: **Moonstruck, True Romance, Johnny English, American Beauty, The Princess Bride, Top Gun, Double Platinum, Rising Sun, Dead Poets Society, Waiting for Guffman**		
# training samples	2	4	10
Top 10 recommended movies by CTR	Swordfish	**Pulp Fiction**	**Best in Snow**
	A Fish Called Wanda	A Clockwork Orange	**Chocolat**
	Terminator 2	Being John Malkovich	**Good Will Hunting**
	A Clockwork Orange	**Raising Arizona**	**Monty Python and the Holy Grail**
	Sling Blade	Sling Blade	Being John Malkovich
	Bridget Jones's Diary	Swordfish	**Raising Arizona**
	Raising Arizona	A Fish Called Wanda	The Graduate
	A Streetcar Named Desire	Saving Grace	Swordfish
	The Untouchables	The Graduate	Tootsie
	The Full Monty	Monster's Ball	Saving Private Ryan
# training samples	2	4	10
Top 10 recommended movies by CDL	Snatch	**Pulp Fiction**	**Good Will Hunting**
	The Big Lebowski	Snatch	**Best in Show**
	Pulp Fiction	**The Usual Suspect**	**The Big Lebowski**
	Kill Bill	Kill Bill	**A Few Good Men**
	Raising Arizona	Momento	**Monty Python and the Holy Grail**
	The Big Chill	**The Big Lebowski**	**Pulp Fiction**
	Tootsie	**One Flew Over the Cuckoo's Nest**	The Matrix
	Sense and Sensibility	As Good as It Gets	**Chocolat**
	Sling Blade	**Goodfellas**	**The Usual Suspect**
	Swinger	The Matrix	**CaddyShack**

Precision Comparison (P=10, dense)

	CDL	CTR
2 samples	30%	20%
4 samples	50%	20%
10 samples	90%	50%

CDL provides more **accurate** recommendation

Experiments

Complexity Analysis

Notations

u_i: latent user vector

v_j: latent item vector

K: dimension of learned representation

I, J: number of users, items

S: size of vocabulary

K_1: dimension of output in the 1st layer

Update rules

$$\mathbf{u}_i \leftarrow (\mathbf{V}\mathbf{C}_i\mathbf{V}^T + \lambda_u \mathbf{I}_K)^{-1}\mathbf{V}\mathbf{C}_i\mathbf{R}_i$$

$$\mathbf{v}_j \leftarrow (\mathbf{U}\mathbf{C}_j\mathbf{U}^T + \lambda_v \mathbf{I}_K)^{-1}(\mathbf{U}\mathbf{C}_j\mathbf{R}_j + \lambda_v f_e(\mathbf{X}_{0,j*}, \mathbf{W}^+)^T)$$

complexity of updating u_i: $O(K^2 J + K^3)$

complexity of updating v_j: $O(K^2 I + K^3 + SK_1)$

complexity of updating all weights, biases: $O(JSK_1)$

complexity of complete epoch:

$O(K^2 I^2 + K^2 J^2 + K^3 + JSK_1)$

Conclusions

AutoRec

- apply Autoencoder for collaborative filtering
- efficient, effective model
- Nonlinear latent representations are available

CDL

- 1st hierarchical Bayesian model
- Bridge the gap between RS and state-of-the-art deep learning models
- Scalable model
- State-of-the-art performance by jointly performing representation learning and collaborative filtering

THANK YOU