Recommender Systems III

Deep Learning Approaches

Recall: Basics of Recommender Systems

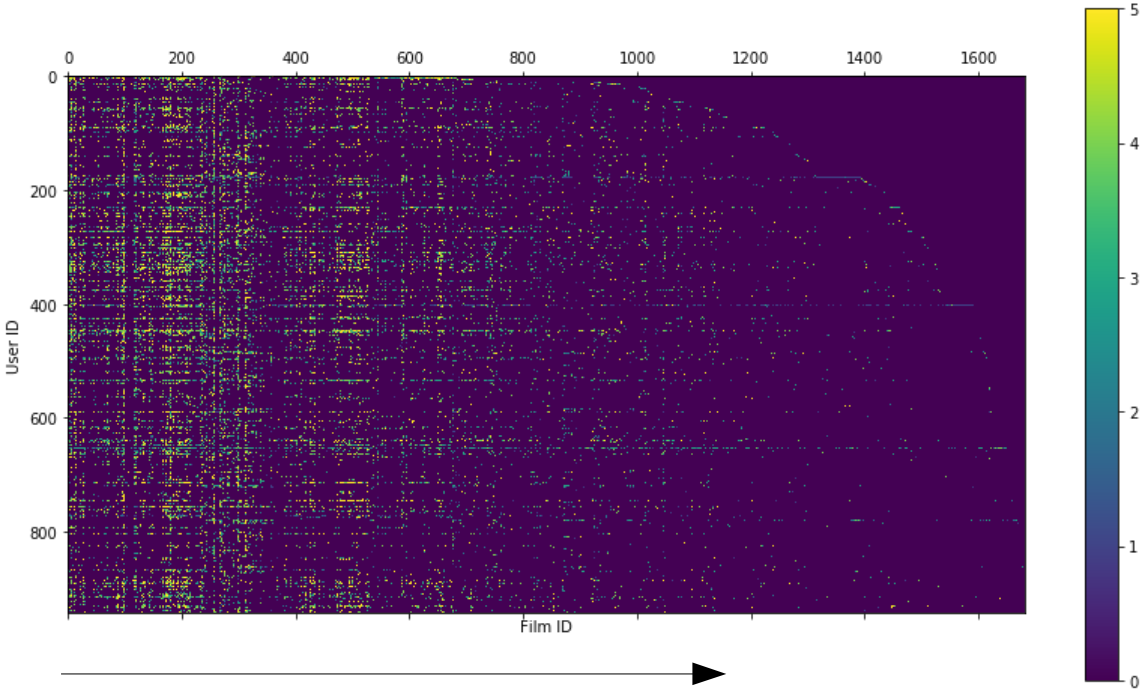
Hochschule Offenburg
offenburg.university

Complexity of Collaborative Filter Approach

build Matrix

$$R \text{ of size } n \times m \text{ with context measure } r_{ij}, \text{ for } i \in 1 \dots n, j \in 1 \dots m$$

User ID



Item ID

Basics of Recommender Systems

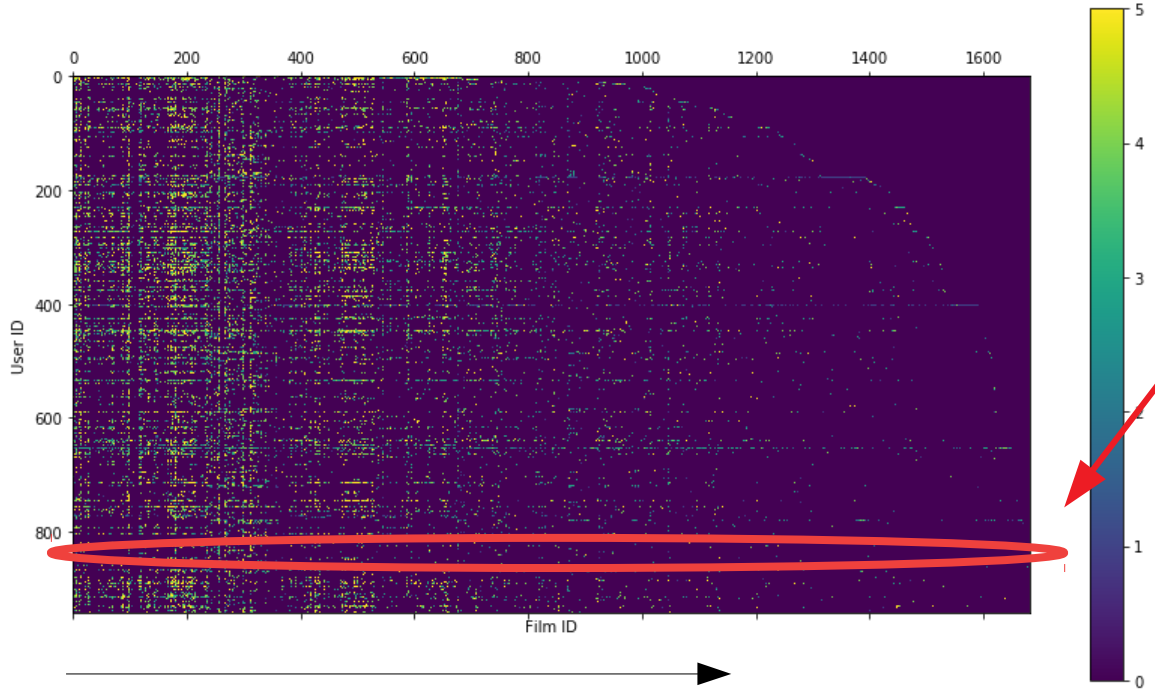
Hochschule Offenburg
offenburg.university

Number of Users n

build Matrix

R of size $n \times m$ with context measure r_{ij}, for $i \in 1 \ldots n, j \in 1 \ldots m$

User ID



User vector of Size m

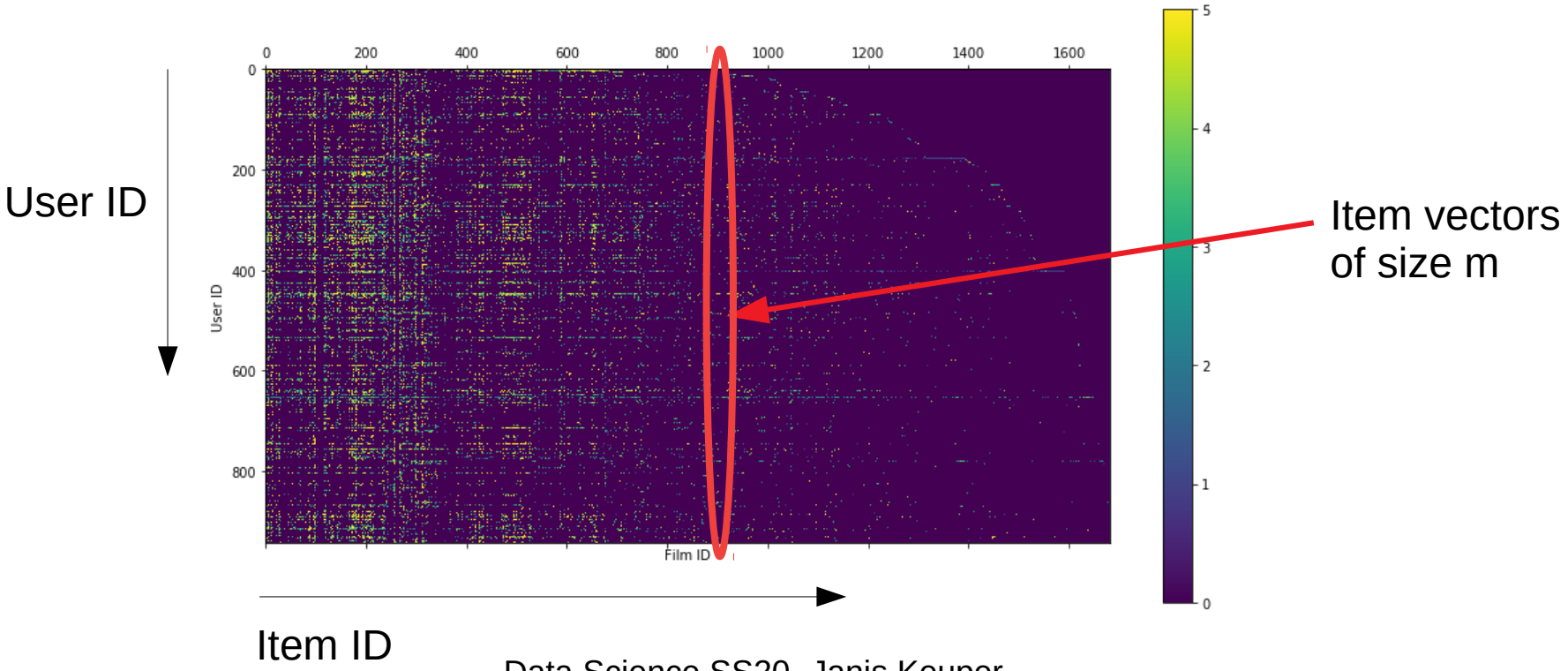
Item ID

Basics of Recommender Systems

Number of items m

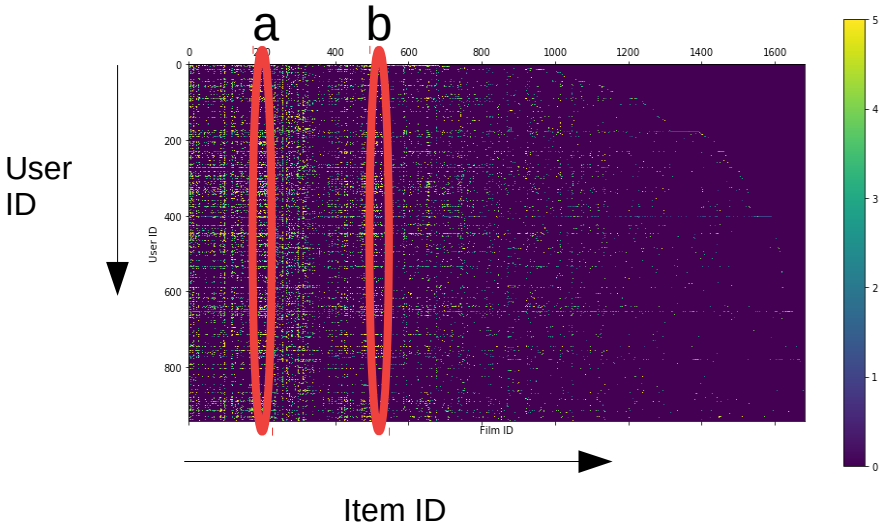
build Matrix

R of size $n \times m$ with context measure r_{ij}, for $i \in 1 \ldots n, j \in 1 \ldots m$



Basics of Recommender Systems

Compute Distances between items



Items are encoded in column vectors

→ distance between items: vector distance

$$d_{cos}(\vec{a}, \vec{b}) := \frac{< \vec{a}, \vec{b} >}{|\vec{a}||\vec{b}|}$$

e.g. cosine distance

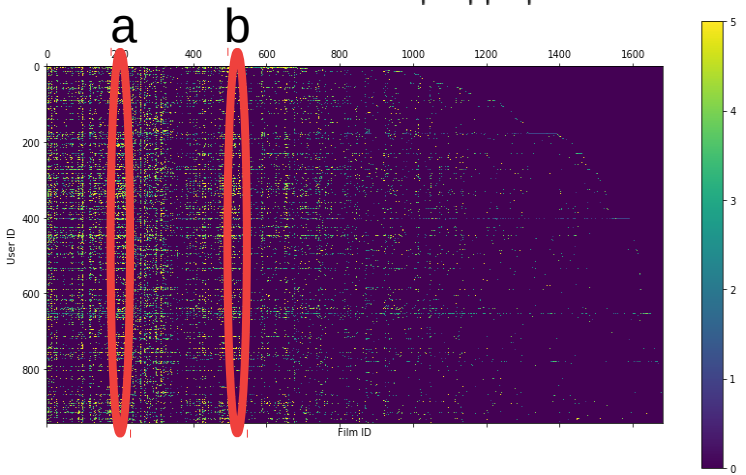
Problem I: R very large and very sparse (e.g. Amazon: 300 million items!)
Problem II: R is changing over time → need to recompute

Basics of Recommender Systems

Approximate R by SVD

$$\tilde{R} = \tilde{U}\tilde{\Sigma}\tilde{V}^*$$

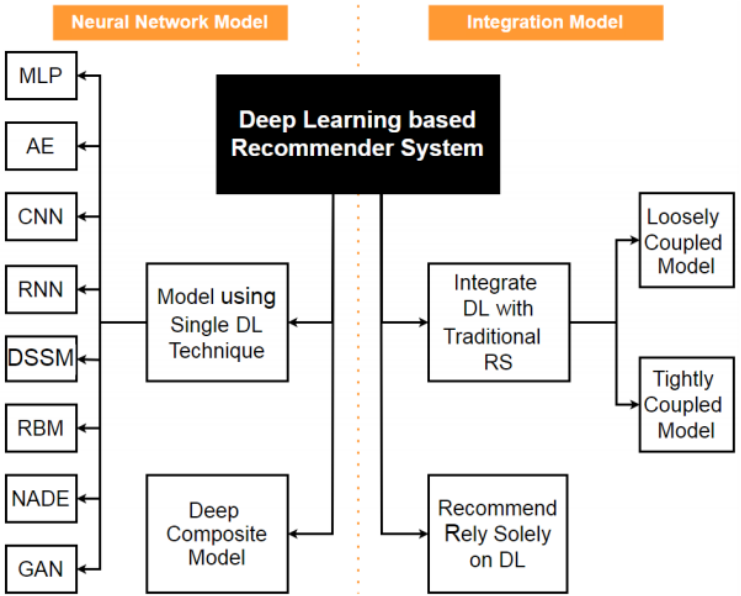
$$d_{cos}(\vec{a}, \vec{b}) := \frac{< \vec{a}, \vec{b} >}{|\vec{a}||\vec{b}|}$$



Solution:

- We do not need the full R Matrix at any time
  → need only item vectors to compute distance
- Use a vector by vector SVD (next slide)
- Store and update vectors independently, only recompute for those items/users were changes happen
- Compute distance for every query, based on SVD approximated vectors a and b

Deep Learning for Recommender Systems



Categorization of existing DL RS approaches

Deep Learning based Recommender System: A Survey and New Perspectives

SHUAI ZHANG, University of New South Wales
LINA YAO, University of New South Wales
AIXIN SUN, Nanyang Technological University

With the ever-growing volume, complexity and dynamicity of online information, recommender system has been an effective key solution to overcome such information overload. In recent years, deep learning's revolutionary advances in speech recognition, image analysis and natural language processing have gained significant attention. Meanwhile, recent studies also demonstrate its effectiveness in coping with information retrieval and recommendation tasks. Applying deep learning techniques into recommender system has been gaining momentum due to its state-of-the-art performances and high-quality recommendations. In contrast to traditional recommendation models, deep learning provides a better understanding of user's demands, item's characteristics and historical interactions between them.

This article aims to provide a comprehensive review of recent research efforts on deep learning based recommender systems towards fostering innovations of recommender system research. A taxonomy of deep learning based recommendation models is presented and used to categorize the surveyed articles. Open problems are identified based on the analytics of the reviewed works and discussed potential solutions.

CCS Concepts: •**Information systems** → **Recommender systems;**

Additional Key Words and Phrases: Recommender System; Deep Learning; Survey

ACM Reference format:
Shuai Zhang, Lina Yao, and Aixin Sun. 2017. Deep Learning based Recommender System: A Survey and New Perspectives. *ACM J. Comput. Cult. Herit.* 1, 1, Article 35 (July 2017), 35 pages.
DOI: 0000001.0000001

1 INTRODUCTION

The explosive growth of information available online frequently overwhelms users. Recommender system (RS) is a useful information filtering tool for guiding users in a personalized way of discovering products or services they might be interested in from a large space of possible options. Recommender system has been playing a more vital and essential role in various information access systems to boost business and facilitate decision-making process.

In general, the recommendation lists are generated based on user preferences, item features, user-item past interactions and some other additional information such as temporal and spatial data. Recommendation models are mainly categorized into collaborative filtering, content-based recommender system and hybrid recommender system based on the types of input data [1]. However, these models have their own limitations in dealing with data sparsity and cold-start problems, as well as balancing the recommendation qualities in terms of different evaluation metrics[1, 6, 74, 110].

Author's addresses: S. Zhang and L. Yao, School of Computer Science and Engineering, University of New South Wales, Sydney, NSW, 2052; emails: shuai.zhang@student.unsw.edu.au; lina.yao@unsw.edu.au; A. Sun, School of Computer Science and Engineering, Nanyang

2017

Deep Learning for Recommender Systems

Hochschule Offenburg
offenburg.university

		Recommend Rely Solely on Deep Learning	Integrate Deep Learning with Traditional Recommender System	
			Tightly Coupled Model	*Loosely Coupled Model*
Model Using Single Deep Learning Technique	*MLP (15)*	[46], [133], [2], [111] [118], [38], [9], [25] [12]	[10], [65], [17], [35] [39]	[66]
	AE (25)	[82], [90], [99], [129] [98], [135], [144], [102] [84]	[113], [63], [62], [24] [112] , [145], [3], [136]	[139], [122], [123], [106] [7], [146], [107], [22]
	CNN (17)	[32], [79], [120]	[51], [52], [140], [109] [91], [69]	[15], [93], [143], [73] [36], [37], [124]. [117]
	RNN (22)	[4], [53], [94], [18] [64], [104], [126], [49] [41], [128], [127], [40] [101], [103], [77], [130] [125], [23]	[19], [105], [81]	[96]
	DSSM (3)	[26], [8], [132]	*	*
	RBM (7)	[89], [30], [70], [131] [47]	*	[48], [119]
	NADE (2)	[142], [141]	*	*
	GAN (1)	*	[116]	*
Deep Composite Model (10)		[59], [85], [138], [97] [25], [61], [58], [28]	[137], [114]	*

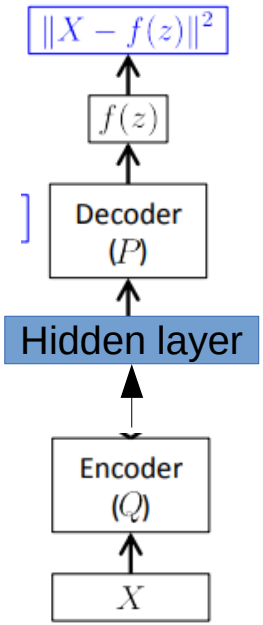
Notation for the following methods:

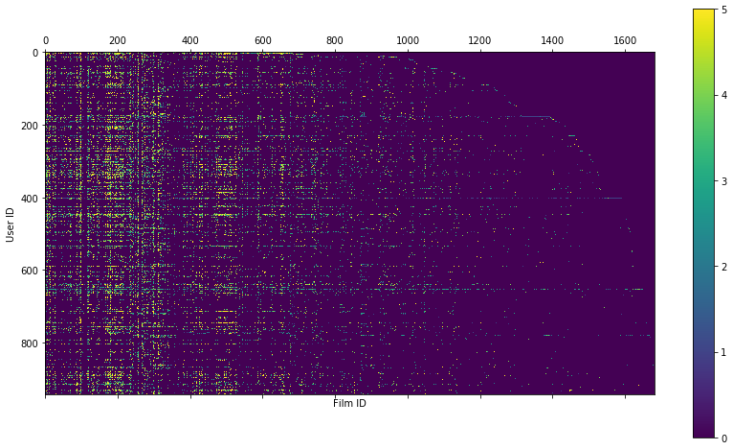
Notation	Description	Notation	Description
R	Rating matrix	$\hat{R}$	Predicted rating matrix
M	Number of users	N	Number of items
r_{ui}	Rating of item i given by user u	$\hat{r}_{ui}$	Predicted rating of item i given by user u
$\mathbf{r}^{(i)}$	Partial observed vector for item i	$\mathbf{r}^{(u)}$	Partial observed vector for user u
U	User latent factor	V	Item latent factor
k	The size of latent factor	K	The maximum value of rating (explicit)
$\mathcal{O}$, $\mathcal{O}^-$	Observed, Unobserved rating set	I	Indicator, $I_{ui} = 1$ if $r_{ui} \neq 0$ otherwise $I_{ui} = 0$
W_*	Weight matrices for neural network	b_*	Bias terms for neural network

Autoencoder Recommender Systems

Basic ideas of AE approach:

- use AE on **user** **or** **item** vectors of R



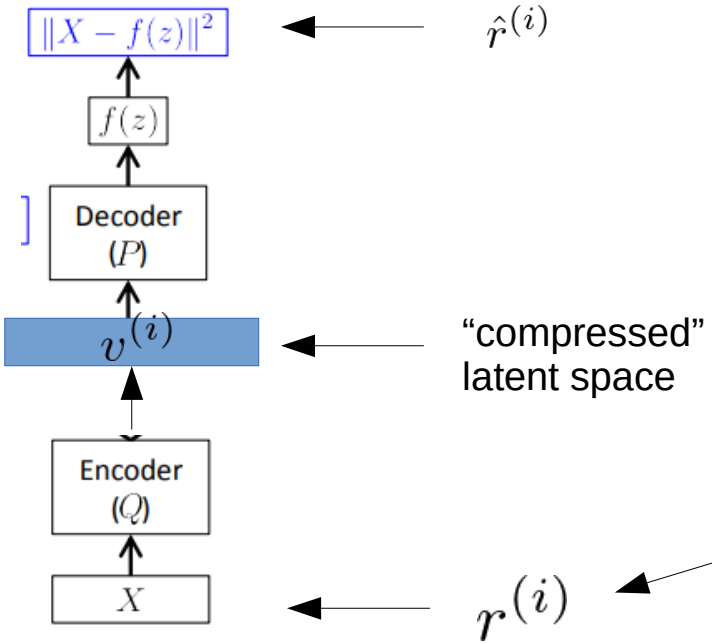


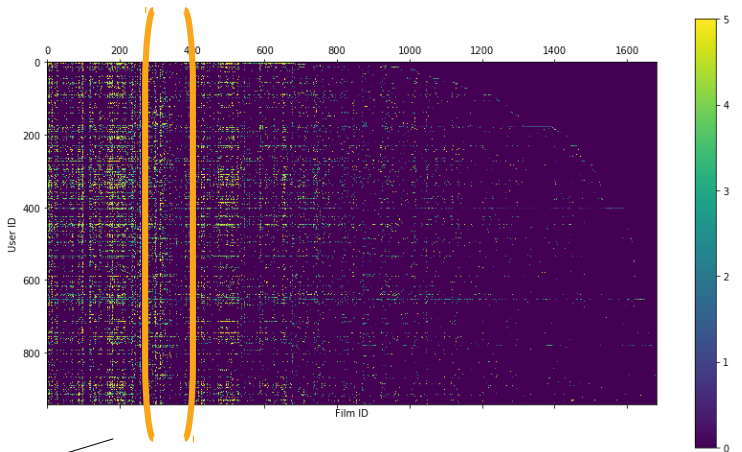
Rating Matrix *R*

Autoencoder Recommender Systems

Basic ideas of AE approach:

- use AE on **user or item** vectors of R



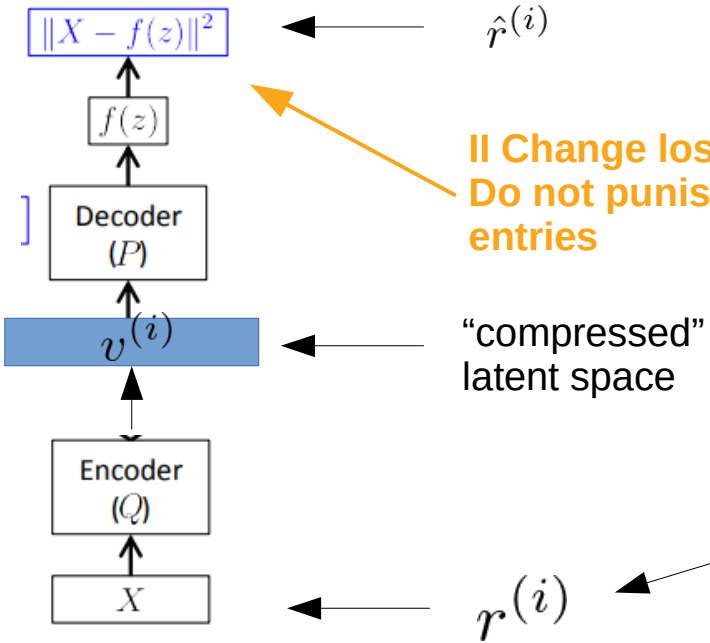


Rating Matrix *R*

Autoencoder Recommender Systems

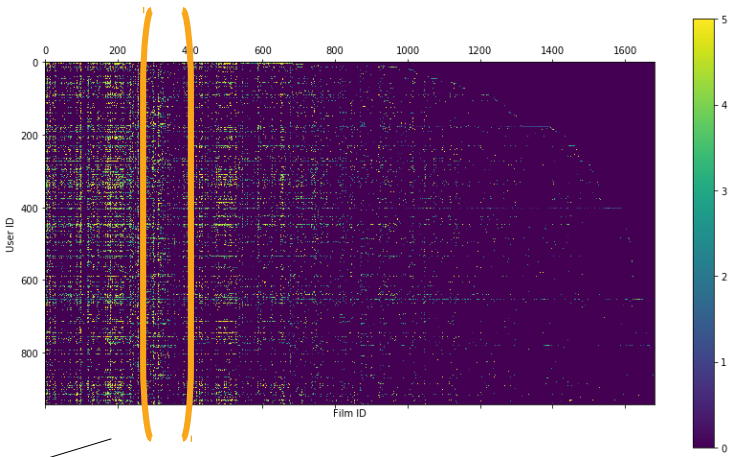
Basic ideas of AE approach:

I use AE on **user or item** vectors of R



II Change loss function:
Do not punish filling of zero entries

"compressed"
latent space

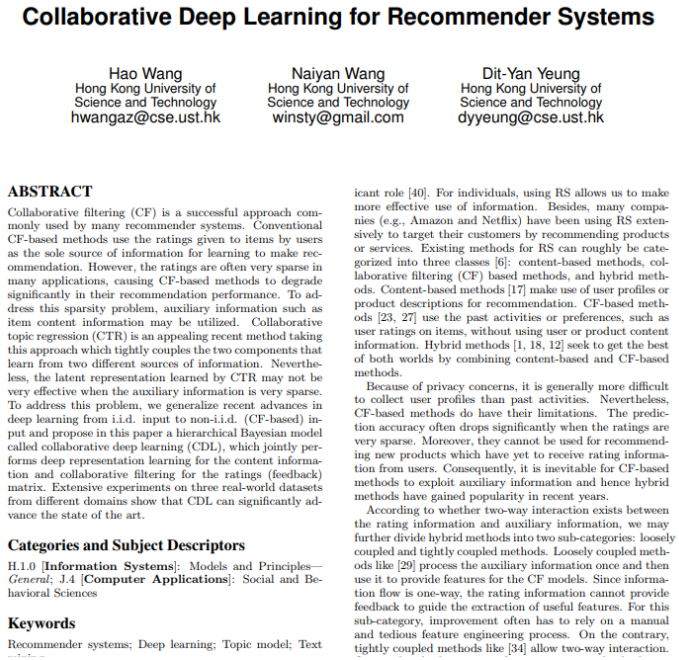


Rating Matrix *R*

Autoencoder Recommender Systems

Hochschule Offenburg
offenburg.university

More sophisticated AE approaches
Also include user information ...



Collaborative Deep Learning for Recommender Systems

Hao Wang
Hong Kong University of Science and Technology
hwangaz@cse.ust.hk

Naiyan Wang
Hong Kong University of Science and Technology
winsty@gmail.com

Dit-Yan Yeung
Hong Kong University of Science and Technology
dyyeung@cse.ust.hk

ABSTRACT

Collaborative filtering (CF) is a successful approach commonly used by many recommender systems. Conventional CF-based methods use the ratings given to items by users as the sole source of information for learning to make recommendation. However, the ratings are often very sparse in many applications, causing CF-based methods to degrade significantly in their recommendation performance. To address this sparsity problem, auxiliary information such as item content information may be utilized. Collaborative topic regression (CTR) is an appealing recent method taking this approach which tightly couples the two components that learn from two different sources of information. Nevertheless, the latent representation learned by CTR may not be very effective when the auxiliary information is very sparse. To address this problem, we generalize recent advances in deep learning from i.i.d. input to non-i.i.d. (CF-based) input and propose in this paper a hierarchical Bayesian model called collaborative deep learning (CDL), which jointly performs deep representation learning for the content information and collaborative filtering for the ratings (feedback) matrix. Extensive experiments on three real-world datasets from different domains show that CDL can significantly advance the state of the art.

Categories and Subject Descriptors

H.1.0 [**Information Systems**]: Models and Principles—*General*; J.4 [**Computer Applications**]: Social and Behavioral Sciences

Keywords

Recommender systems; Deep learning; Topic model; Text

icant role [40]. For individuals, using RS allows us to make more effective use of information. Besides, many companies (e.g., Amazon and Netflix) have been using RS extensively to target their customers by recommending products or services. Existing methods for RS can roughly be categorized into three classes [6]: content-based methods, collaborative filtering (CF) based methods, and hybrid methods. Content-based methods [17] make use of user profiles or product descriptions for recommendation. CF-based methods [23, 27] use the past activities or preferences, such as user ratings on items, without using user or product content information. Hybrid methods [1, 18, 12] seek to get the best of both worlds by combining content-based and CF-based methods.

Because of privacy concerns, it is generally more difficult to collect user profiles than past activities. Nevertheless, CF-based methods do have their limitations. The prediction accuracy often drops significantly when the ratings are very sparse. Moreover, they cannot be used for recommending new products which have yet to receive rating information from users. Consequently, it is inevitable for CF-based methods to exploit auxiliary information and hence hybrid methods have gained popularity in recent years.

According to whether two-way interaction exists between the rating information and auxiliary information, we may further divide hybrid methods into two sub-categories: loosely coupled and tightly coupled methods. Loosely coupled methods like [29] process the auxiliary information once and then use it to provide features for the CF models. Since information flow is one-way, the rating information cannot provide feedback to guide the extraction of useful features. For this sub-category, improvement often has to rely on a manual and tedious feature engineering process. On the contrary, tightly coupled methods like [34] allow two-way interaction.

CNN based Recommender Systems

Hochschule Offenburg
offenburg.university

Two basic variants of CNN approach:

- Use CNNs for item (and user) content feature extraction

Examples:

- Movie similarity by image similarity of posters

CNN based distance measure

https://www.designmantic.com/blog/movie-moods-in-typography/

CNN based Recommender Systems

Hochschule Offenburg
offenburg.university

Basic variants of CNN approach:

Use CNNs for item (and user) content feature extraction

Examples:

- Movie similarity by image similarity of posters

- Clothing and fashion items

CNN based distance measure

CNN based Recommender Systems

Hochschule Offenburg
offenburg.university

Basic variants of CNN approach:

Use CNNs for item (and user) content feature extraction

Examples:

- Movie similarity by image similarity of posters

- Clothing and fashion items

CNN based distance measure

CNN and AE approaches can be directly combined
or CNN and SVD ...

RNN based Recommender Systems

Hochschule Offenburg
offenburg.university

Using RNNs for text feature extraction:

I User features: reviews and comments by a user

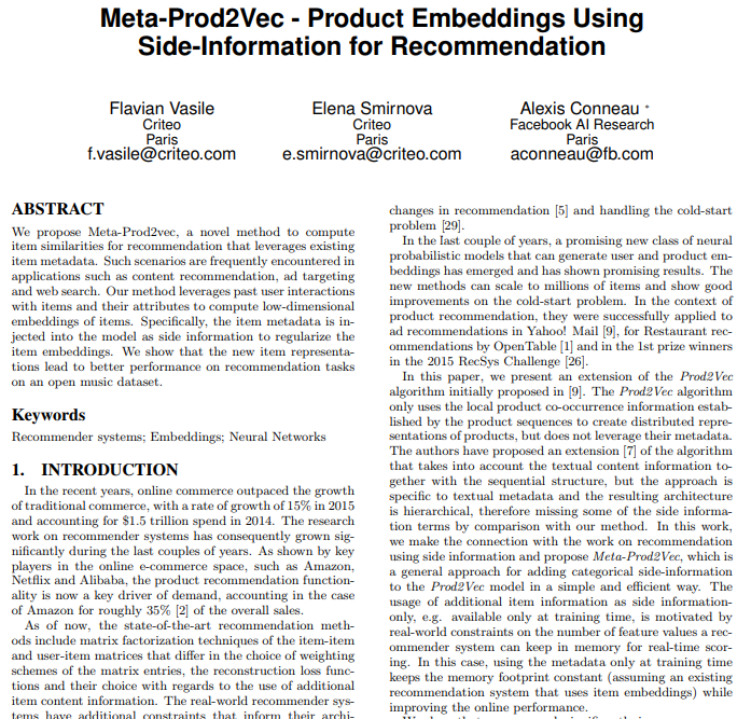
II Item features: text description and reviews of item

Embedding based Recommender Systems

Hochschule Offenburg
offenburg.university

Embeddings

- word2vec scheme for product embeddings



Meta-Prod2Vec - Product Embeddings Using Side-Information for Recommendation

Flavian Vasile
Criteo
Paris
f.vasile@criteo.com

Elena Smirnova
Criteo
Paris
e.smirnova@criteo.com

Alexis Conneau *
Facebook AI Research
Paris
aconneau@fb.com

ABSTRACT

We propose Meta-Prod2vec, a novel method to compute item similarities for recommendation that leverages existing item metadata. Such scenarios are frequently encountered in applications such as content recommendation, ad targeting and web search. Our method leverages past user interactions with items and their attributes to compute low-dimensional embeddings of items. Specifically, the item metadata is injected into the model as side information to regularize the item embeddings. We show that the new item representations lead to better performance on recommendation tasks on an open music dataset.

Keywords

Recommender systems; Embeddings; Neural Networks

1. INTRODUCTION

In the recent years, online commerce outpaced the growth of traditional commerce, with a rate of growth of 15% in 2015 and accounting for $1.5 trillion spend in 2014. The research work on recommender systems has consequently grown significantly during the last couples of years. As shown by key players in the online e-commerce space, such as Amazon, Netflix and Alibaba, the product recommendation functionality is now a key driver of demand, accounting in the case of Amazon for roughly 35% [2] of the overall sales.

As of now, the state-of-the-art recommendation methods include matrix factorization techniques of the item-item and user-item matrices that differ in the choice of weighting schemes of the matrix entries, the reconstruction loss functions and their choice with regards to the use of additional item content information. The real-world recommender systems have additional constraints that inform their architectures changes in recommendation [5] and handling the cold-start problem [29].

In the last couple of years, a promising new class of neural probabilistic models that can generate user and product embeddings has emerged and has shown promising results. The new methods can scale to millions of items and show good improvements on the cold-start problem. In the context of product recommendation, they were successfully applied to ad recommendations in Yahoo! Mail [9], for Restaurant recommendations by OpenTable [1] and in the 1st prize winners in the 2015 RecSys Challenge [26].

In this paper, we present an extension of the *Prod2Vec* algorithm initially proposed in [9]. The *Prod2Vec* algorithm only uses the local product co-occurrence information established by the product sequences to create distributed representations of products, but does not leverage their metadata. The authors have proposed an extension [7] of the algorithm that takes into account the textual content information together with the sequential structure, but the approach is specific to textual metadata and the resulting architecture is hierarchical, therefore missing some of the side information terms by comparison with our method. In this work, we make the connection with the work on recommendation using side information and propose *Meta-Prod2Vec*, which is a general approach for adding categorical side-information to the *Prod2Vec* model in a simple and efficient way. The usage of additional item information as side information-only, e.g. available only at training time, is motivated by real-world constraints on the number of feature values a recommender system can keep in memory for real-time scoring. In this case, using the metadata only at training time keeps the memory footprint constant (assuming an existing recommendation system that uses item embeddings) while improving the online performance.

Frameworks for Deep Recommender Systems

Hochschule Offenburg
offenburg.university

Tensorflow might not be the right framework …

Check out Amazons's

Deep Scalable Sparse Tensor Network Engine (DSSTNE)

https://github.com/amzn/amazon-dsstne

Optimized for DL on very sparse and very large data ...

Provides very good Spark integration:

https://aws.amazon.com/blogs/big-data/generating-recommendations-at-amazon-scale-with-apache-spark-and-amazon-dsstne/