Deep Learning Based Recommendation Systems

November 2021, Thessaloniki

Postgraduate Program

Department of Applied Informatics , University of Macedonia

Table of Contents

Definition of Recommendation System

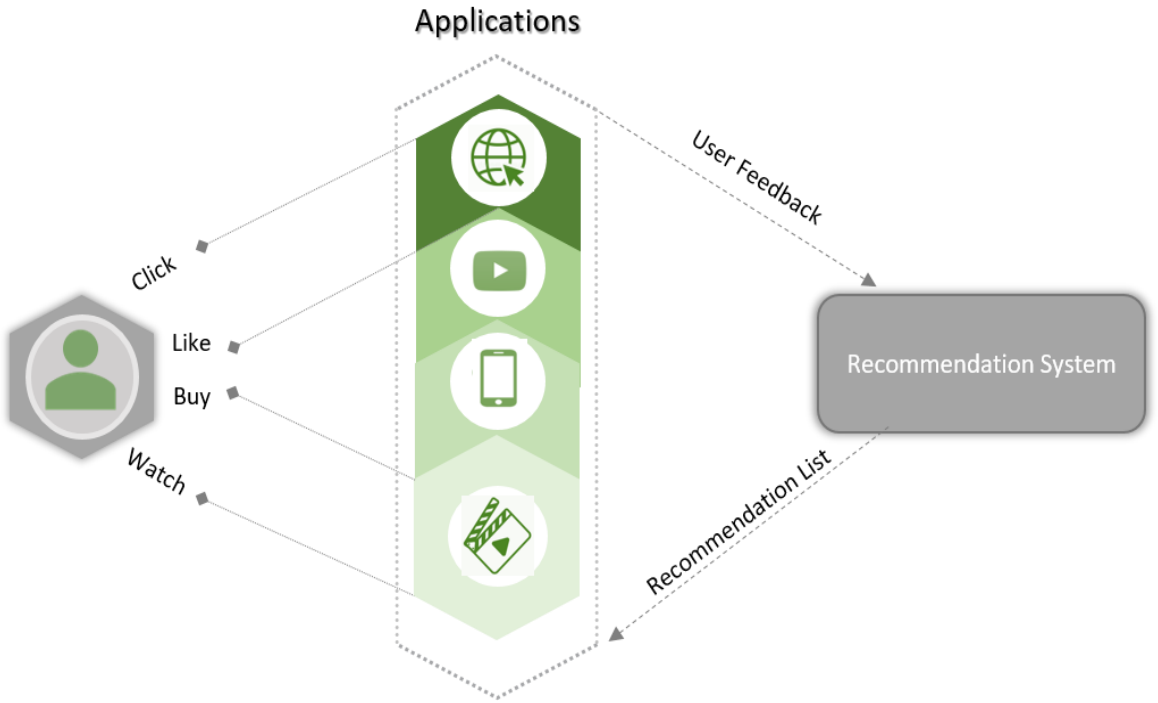
Recommendation systems (RS) are intelligent systems and their methods are intelligent decision-making processes. Their main purpose is to estimate the users' preference for items and then suggest items that would be close to the users' preferences.

RS estimation function R:

On the basis of an initial set of ratings, a recommendation system tries to estimate/predict a rating function R. The rating function R is defined as bellow:

$$(i, j) \rightarrow R$$

where i denotes a user and j denotes an item.



Hierarchy of Recommendation Systems

Model-based :

Use of machine-learning models like Matrix Factorization, PCA, SVD, RBM etc.

Memory-based:

Recommendations based on neighbor's interest (user-based) or items' similarity (item-based)

Collaborative Filtering RS:

Makes predictions and suggestions based on user-item interaction.

Content-based RS :

Tries to guess the features or behavior of a user given the item's features, he/she reacts positively to.

Hybrid-based filtering

Recommendation Systems

Types of feedback

Explicit feedback:

Users' input about their interest in items

- Missing data: Excluding the rating all the other remaining user-item connections are considered as missing data and excluded for the analysis.
- Preference: Explicit feedback has a numerical value that shows.
- Evaluation: There are standard metrics to evaluate the produced predictions such as mean squared error (MSE) or mean absolute error (MAE).

Implicit feedback:

Observation of a user's actions. Browser history, logging, mouse movements, stop of video etc.

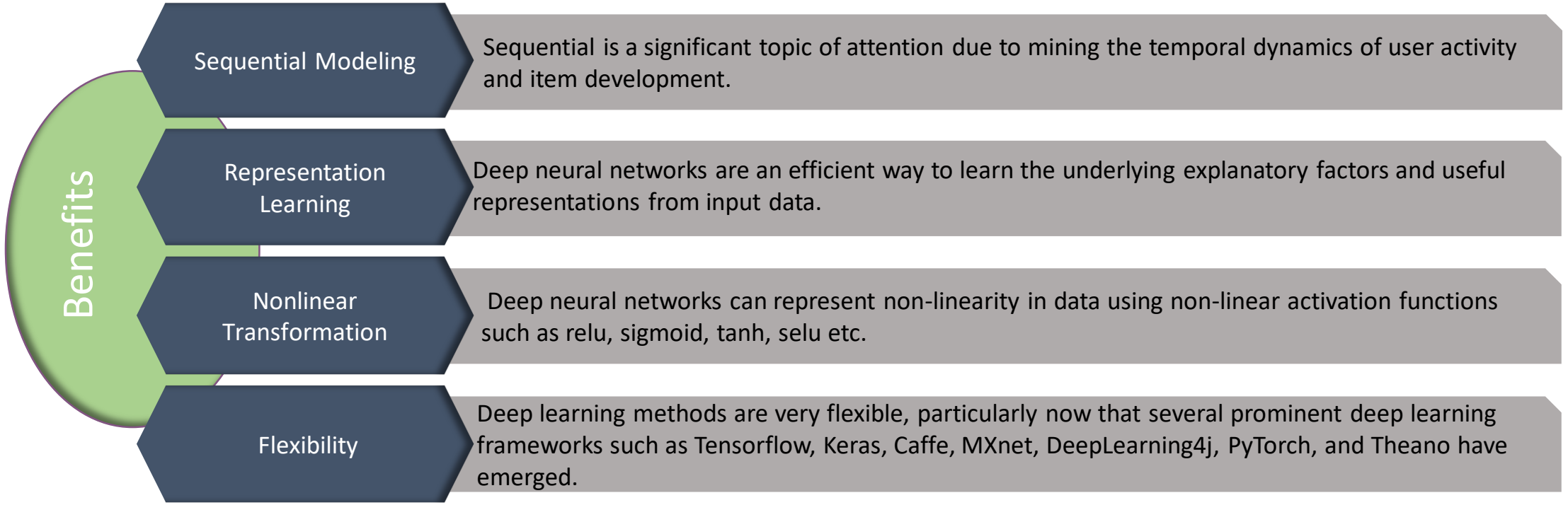
- No negative feedback: Determining which things a user disliked may be difficult
- Noise: We cannot determine users' preferences and real motivations.
- Confidence: Implicit feedback has a numerical value that shows the frequency of activities.

Hybrid feedback:

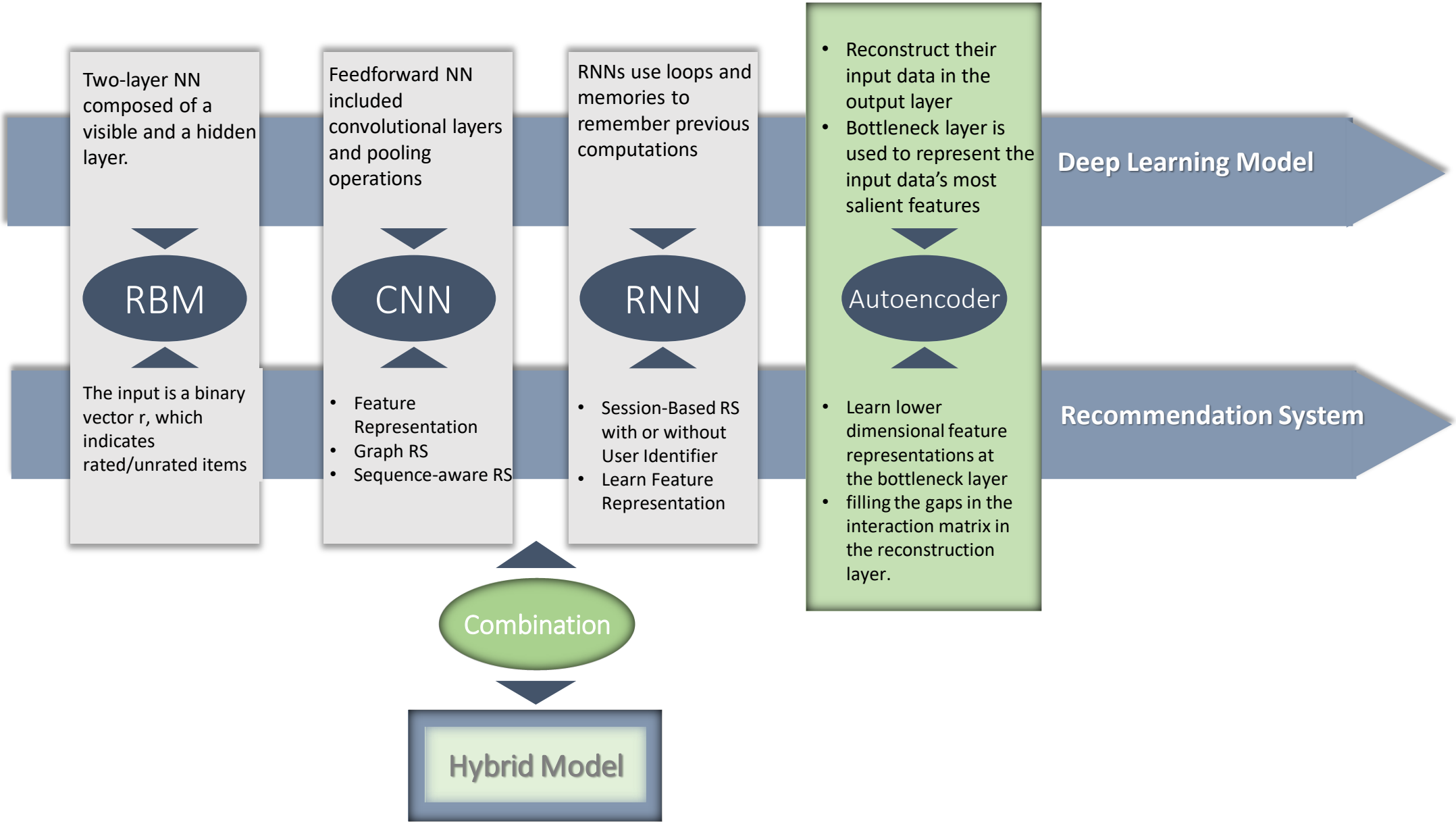
Mix of explicit and implicit feedback

- This method predicts things of interest and taste for users using a mix of numerical ratings and human behavior in order to recover issues like sparsity.

Deep Learning Based Recommendation Systems



Categories of Deep Learning Based Recommendation Systems

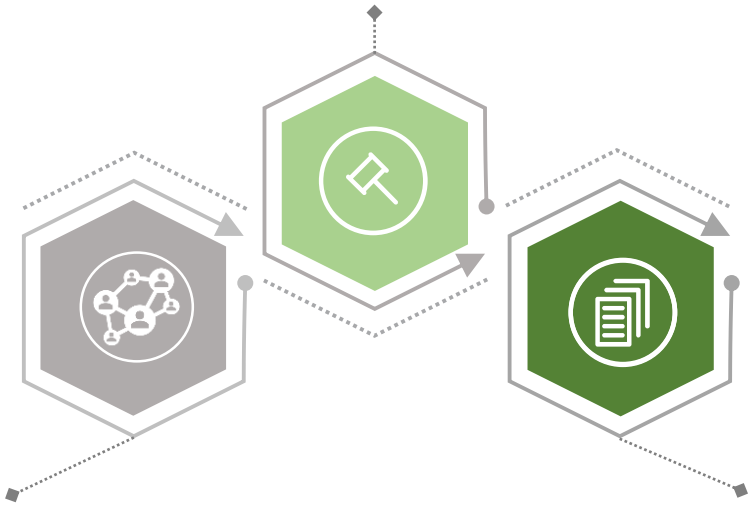


Data Collection- Cleaning- Analysis

Book- Crossing	Total	Explicit	Implicit
Interaction	75.670.906.880	10.104.678.864	10.575.327.973
Ratings	1.149.780	433.671	716.109
Users	278.858	68.012	52.451
Books	271.379	148.572	201.623
Sparsity	99.999%	**99.996%**	99.993%

Focusing on Explicit Ratings, with **high Sparsity** to investigate the **efficiency** of Deep Learning techniques.

Exploring users' and books' useful **side information** and calculating **count and mean of ratings** for each book.

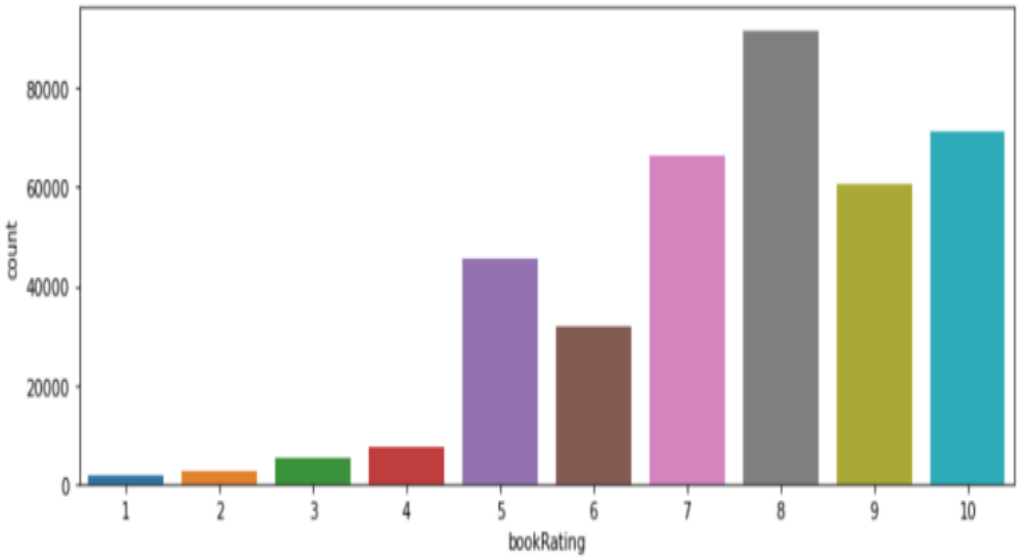


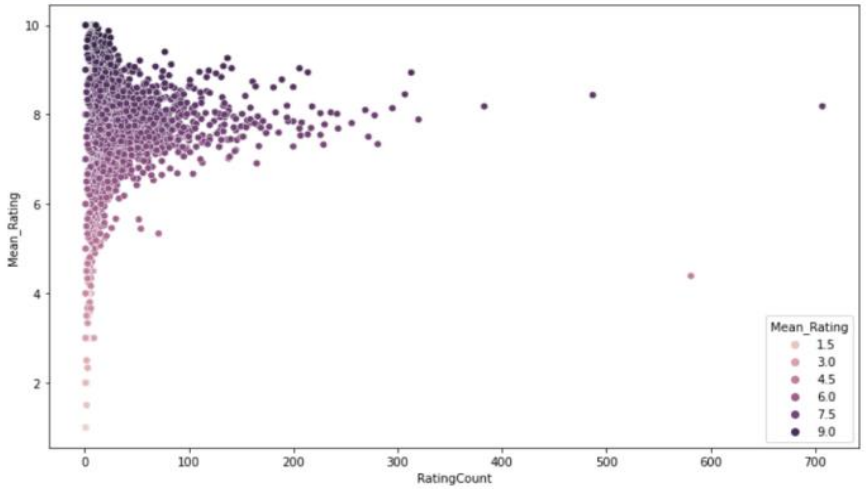
Rating Dataset	Final values
Interaction	57.945.888
Ratings	106.284
Users	6432
Books	9009
Sparsity	**99.816%**

Keeping **books** that are rated by **a minimum number of 5 users and users** that have rated **at least 10 books**. In addition, we also focus on interactions where the corresponding **rating is greater than or equal to 4**.

Exploratory Analysis

- Low Rate (≤3) comprise the 37.72% of total ratings
- The mean of total ratings is quite high (8)





- 99413 books have been rated only once.
- There are 4 ‘popular’ books with more than 350 ratings.

Statistics	Mean Rating	Count of Ratings
Count	149.836	149.836
Mean	7,5275	2,5617
Std	1,7119	7,5056
Min	1	1
25%	6.5	1
50%	8	1
75%	9	2
max	10	707

Evaluation Metrics

Accuracy of predicted Ratings

The percentage of items in the search space for which the algorithm can provide suggestions.

The percentage of relevant items in the test set that appear in the top-10 of the ranking list.

The set diversity for the items in the set of relevant retrieved items

The mean of all relevance scores divided by DCG in ideal order

The sum of all relevance scores in recommendation set

Baseline Recommender Method

Top-10 Recommendations based on Popularity:

Suggest to the user the most popular Books based on the count of ratings.

bookRating	ISBN	bookTitle	bookAuthor	yearOfPublication	publisher
5787	0316666343	The Lovely Bones: A Novel	Alice Sebold	2002	Little, Brown
4108	0385504209	The Da Vinci Code	Dan Brown	2003	Doubleday
3134	0312195516	The Red Tent (Bestselling Backlist)	Anita Diamant	1998	Picador USA
2798	059035342X	Harry Potter and the Sorcerer's Stone (Harry Potter (Paperback))	J. K. Rowling	1999	Arthur A. Levine Books
2595	0142001740	The Secret Life of Bees	Sue Monk Kidd	2003	Penguin Books
2551	0971880107	Wild Animus	Rich Shapero	2004	Too Far
2524	0060928336	Divine Secrets of the Ya-Ya Sisterhood: A Novel	Rebecca Wells	1997	Perennial
2402	0446672211	Where the Heart Is (Oprah's Book Club (Paperback))	Billie Letts	1998	Warner Books
2219	0452282152	Girl with a Pearl Earring	Tracy Chevalier	2001	Plume Books
2179	0671027360	Angels & Demons	Dan Brown	2001	Pocket Star

Singular Value Decomposition (SVD):

allows us to "fill in the gaps" in the rating matrix, estimating the ratings that each user would assign to each item in the dataset.

SVD factors can be represented as an input matrix A into three matrices:

$$A = U \times \Sigma \times V^T$$

$$A = R, Q = U, P^T = \Sigma \times V^T$$

Where A : rating Matrix R ,
Q: item factors ,
P : user factors

k-Nearest Neighbor:

user-based k-NN Basic algorithm is a memory-based method, which uses a group of other users with similar set of preferences in order to recommend items to a user.

Prediction function:

$$\hat{r}_{ui} = \frac{\sum_{u \in N_i^k(u)} sim(u,v) \times r_v i}{\sum_{u \in N_i^k(u)} sim(u,v)}$$

Where
k: number of neighbors,
v: the neighbor of the user,
u: the user,
i: item
sim(u,v): pearson similarity
N: set of the v users for the item i and user u

Results of Baseline Recommendation Methods

Best Match for Hyperparameters

SVD	**Compared values**	**Best value**
Epochs	[10, 20, 30, 40, 50]	30
Learning Rate	[0.001, 0.005, 0.01]	0.005
Regularization lambda	[0.4, 0.6, 0.1]	0.1
RMSE		1.3043
User-based KNN	**Compared values**	**Best value**
Epochs	[10, 20, 30, 40, 50]	40
Learning Rate	[0.001, 0.005, 0.01]	0.001
Regularization lambda	[0.4, 0.6, 0.1]	0.1
RMSE		1.6414

Evaluation of Results

Evaluation Metric	SVD	KNN
RMSE	**1,3043**	1.6414
HitRate@10	0,05	**0.06**
Cumulative HitRate@10 (rating ≥8)	**0,049**	0.039
Novelty	1456.5	1460.21
User Coverage	0.9891	0.9891

AutoRec: Autoencoders meet collaborative filtering

AutoRec neural architecture is described as:

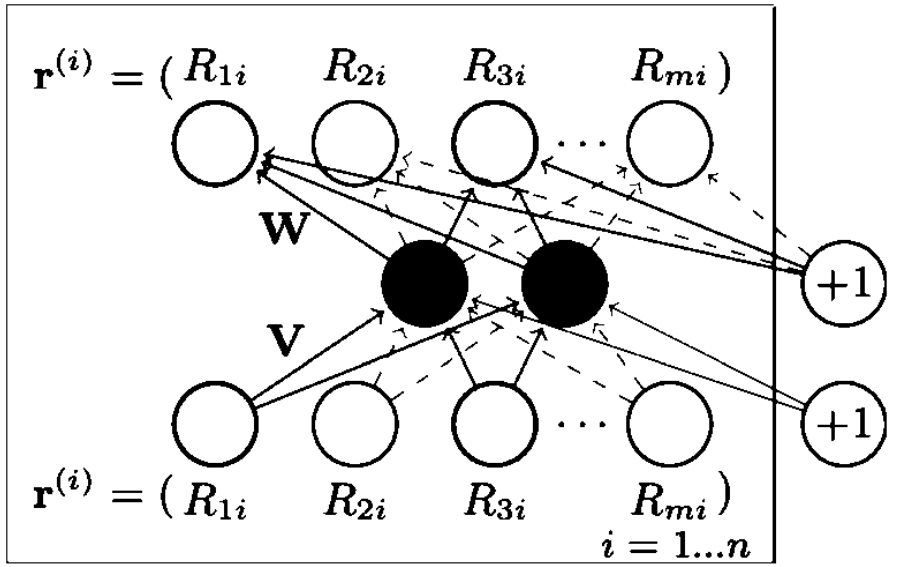
$$h(R_{*i}) = f(W \cdot g(VR_{*i} + \mu) + b)$$

where
$f(\cdot)$: Linear activation function ,
g(·) : Sigmoid activation function,
W and V are weight matrices, μ and b are biases.

As the **objective function** :

$$argmin_{W,V,\mu,b} \sum_{i=1..M} ||R_{*i} - h(R_{*i})||_O^2 + \lambda(||W||_F^2 + ||V||_F^2)$$

where $||\cdot||_O$ denotes that only the contribution of observed ratings is taken into account.

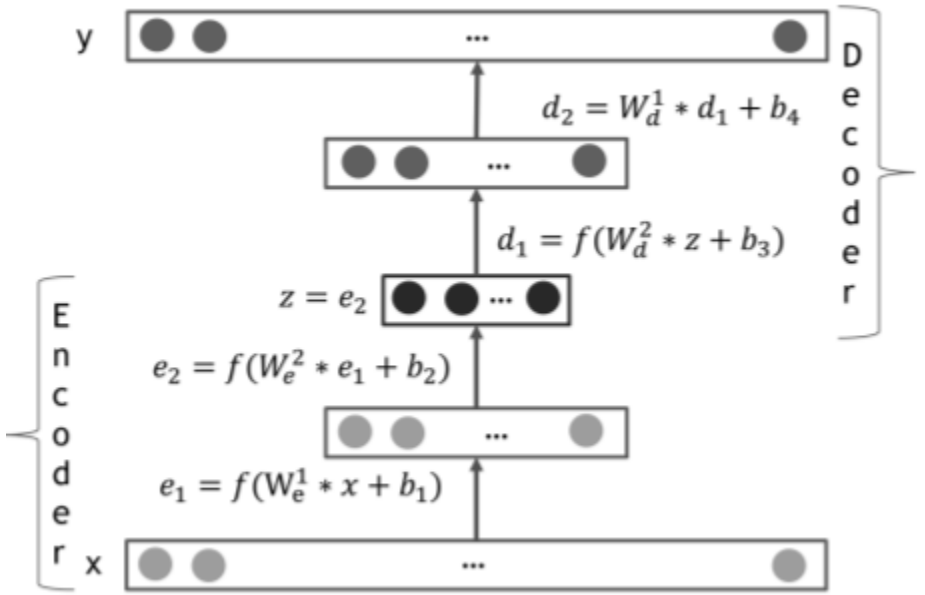


DeepRec: Deep Autoencoder for Recommendation Systems

DeepRec is a feed-forward neural network with fully connected layers computing

$$l = f(Wx + b)$$

where f is a nonlinear activation function
W : the weights of the matrix
b : the biases of the matrix
The decoder's weights W_d^l are constrained to **be equal** to the transposed encoder weights W_e^l from the corresponding layer.

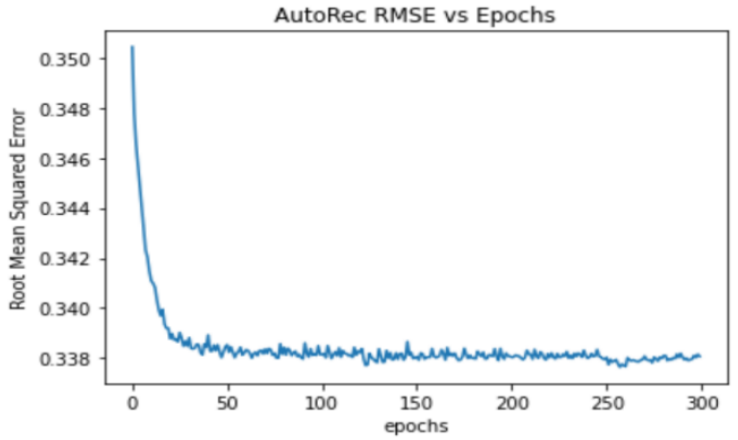


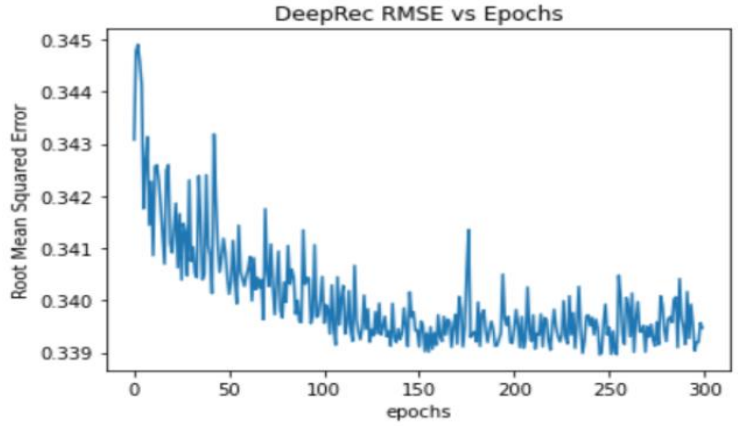
AutoRec and DeepRec Results

Best Match for Hyperparameters

Hyperparameters	Value
AutoRec	
Hidden Layers	500
Regularization l_2	0.0005
Iterations (epochs)	300
Learning Rate	0.0001
Batch size	256
DeepRec	
Hidden Layers	[256, 512, 256]
Regularization l_2	0.001
Iterations (epochs)	300
Learning Rate	0.001
Batch size	256
Drop-out	0.8

RMSE vs Epochs





Neural Collaborative Filtering (NCF) :

A general framework developed to learn the user-item interaction function using neural networks and a probabilistic model.

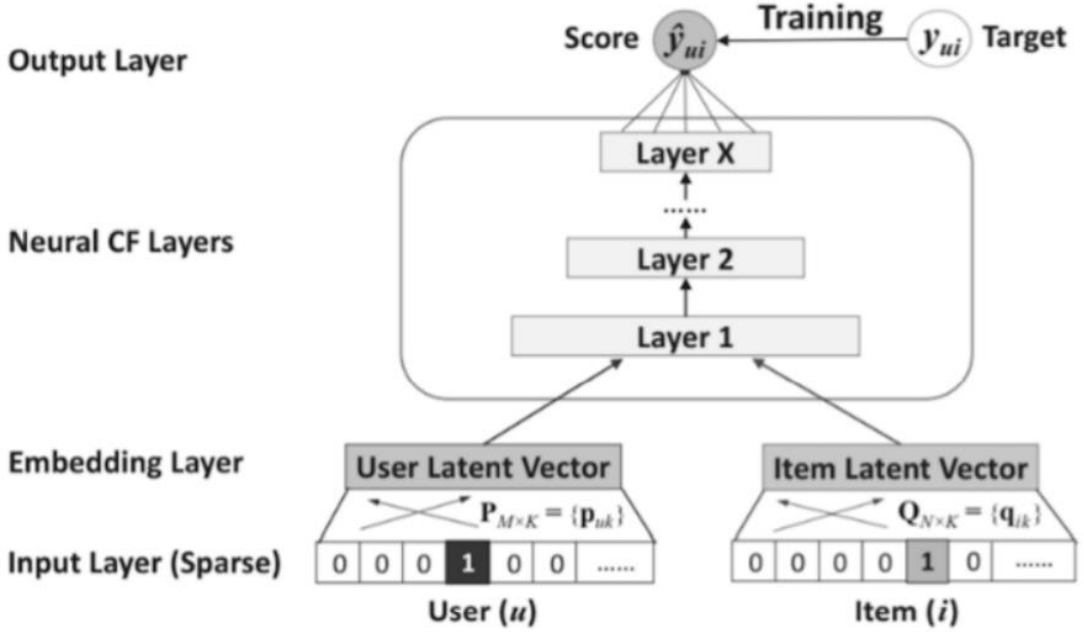
The predictive Model is described by:

$$\hat{y}_{ui} = f(P^T v_u^U Q^T v_i^I | P, Q, \Theta_f)$$

Where Θ consists of model Parameters and f could be described as:

$$f(P^T v_u^U Q^T v_i^I | P, Q, \Theta_f) = \phi_o ut(\phi_X(...(\phi_1(P^T v_u^U Q^T v_i^I))...))$$

Where φ_{out} corresponds to the mapping function for the output layer and φ_{X} to the X-th neural network CF layer.



Learning Model Parameters	
Objective function	▪ Binary cross-entropy for binary feedback ▪ Mean Squared Error, MSE for rating feedback
Optimization function	Adam, SGD

Neural Collaborative filtering

Generalized Matrix Factorization (GMF):

Mapping function is the dot product of u user and i item Vector.

$$\phi_{out}(p_u, q_i) = p_u \cdot q_i$$

Multi Layer Perceptron (MLP):

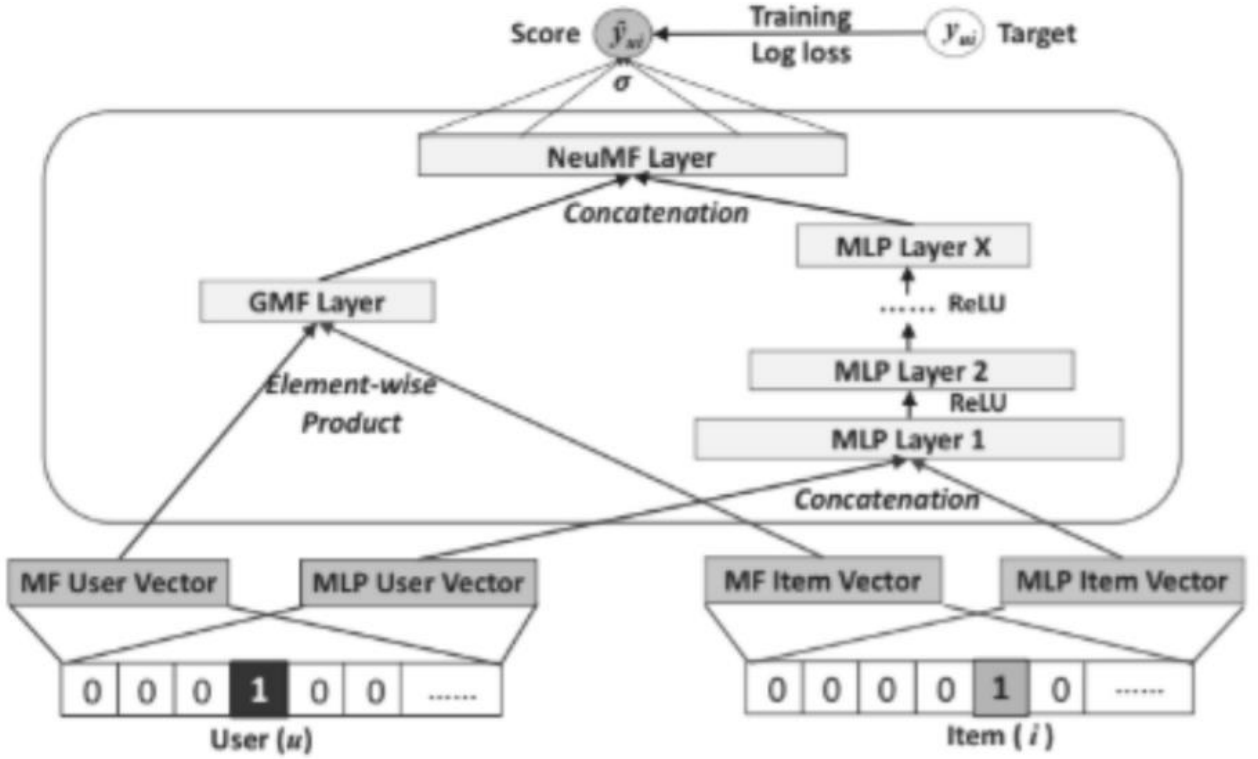
MLP consists of a user-item concatenated vector followed by hidden layers. The model is described by:

$$\hat{y}_{ui} = \sigma(h^T \phi_L(z_{L-1}))$$

Neural Matrix Factorization (NeuMF):

NeuMF is the fusion of GMF and MLP. The model is described by:

$$\hat{y}_{ui} = \sigma(h^T(\phi_{out}^{GMF} \cdot \phi_{out}^{MLP}))$$



Results of GMF, MLP, NeuMF

Best Match for Hyperparameters

Hyperparameters	Value
GMF	
Factors	[8, 16, 32, **64**]
Dropout	[**0.0**, 0.01, 0.001]
Learning Rate	[0.0001, **0.001**, 0.01]
Batch size	[128, **256**, 512]

Hyperparameters	Value
MLP	
Factors	[8, **10**, 16, 32, 64]
Dropout	[**0.0**, 0.01, 0.001]
Learning Rate	[0.0001, **0.001**, 0.01]
Batch size	[128, **256**, 512]

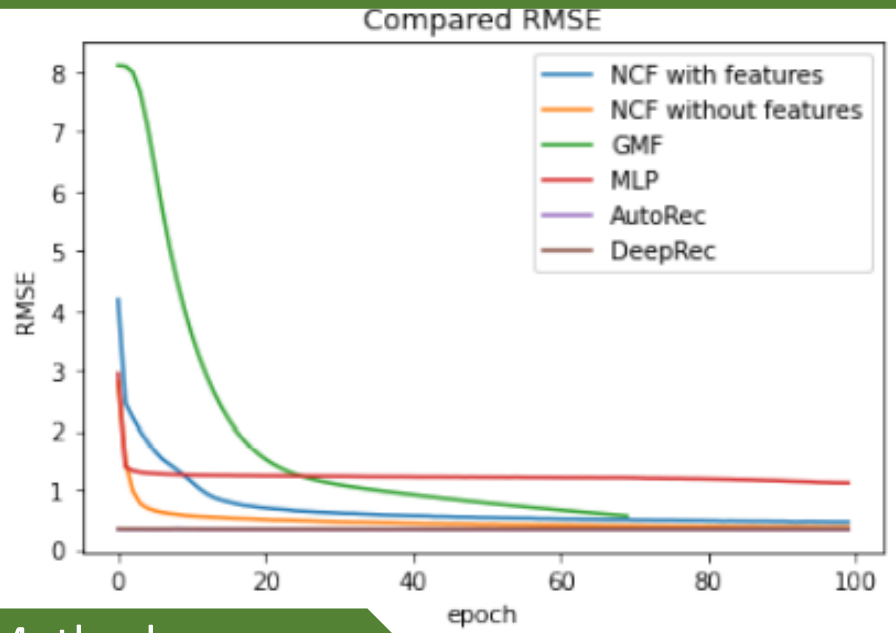
NeuMF	
Factors	[10, 10]
Dropout GMF	[0.2, 0.2, 0.1]
Dropout MLP	[0.2 0.2, 0.1, 0.1]
Learning Rate	0.001
Batch size	256

NeuMF with extra features	
Factors	[10, 10, 64]
Dropout GMF	[0.3, 0.3, 0.2]
Dropout MLP	[0.2 0.2, 0.3, 0.3]
Learning Rate	0.0001
Batch size	256

Comparative Analysis – Evaluation metrics

Comparative Analysis for Deep Learning Models

Model	RMSE
AutoRec	**0.3392**
DeepRec	0.3410
GMF	0.5599
MLP	0.8192
NeuMF	0.3719
Content-based NeuMF	0.4475



Evaluation Metrics NCF vs Baseline Methods

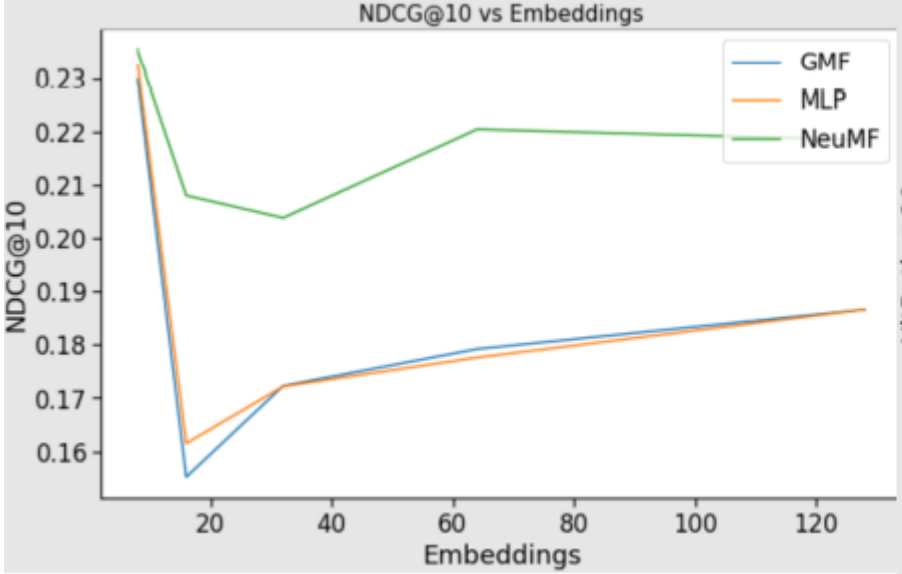
Model	RMSE	HitRate %
KNN	1.6414	6%
SVD	1.3043	5%
GMF	0.5599	24.19%
MLP	0.8192	**27.43**%
NeuMF	**0.3719**	26.99%
Content-based NeuMF	0.4475	25.79%

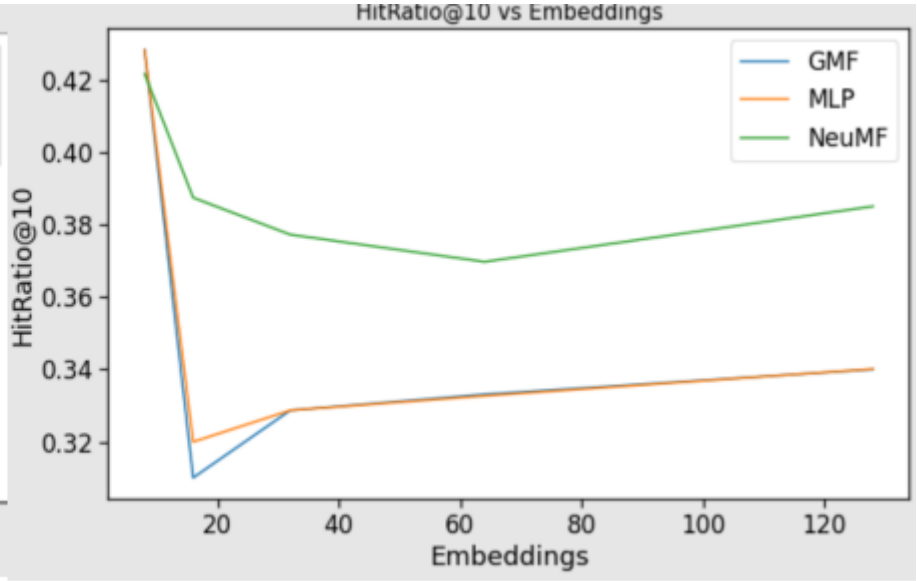
Performance of NCF for observed-unobserved feedback

Evaluation Metrics w.r.t Embeddings

Hit Rate @10			
Embeddings	**GMF**	**MLP**	**NeuMF**
8	0.4282	0.4281	0.4216
16	0.3101	0.3200	0.3874
32	0.3287	0.3287	0.3772
64	0.3331	0.3327	0.3697
128	0.3400	0.3401	0.4049

NDCG@10			
Embeddings	**GMF**	**MLP**	**NeuMF**
8	0.2298	0.2324	0.2350
16	0.1552	0.1615	0.2080
32	0.1723	0.1722	0.2038
64	0.1792	0.1776	0.2204
128	0.1866	0.1866	0.2184



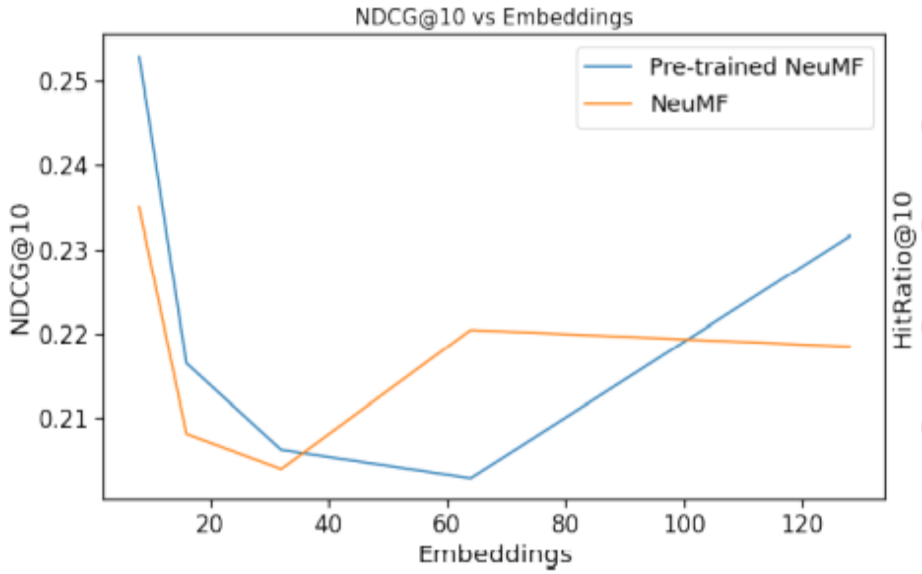


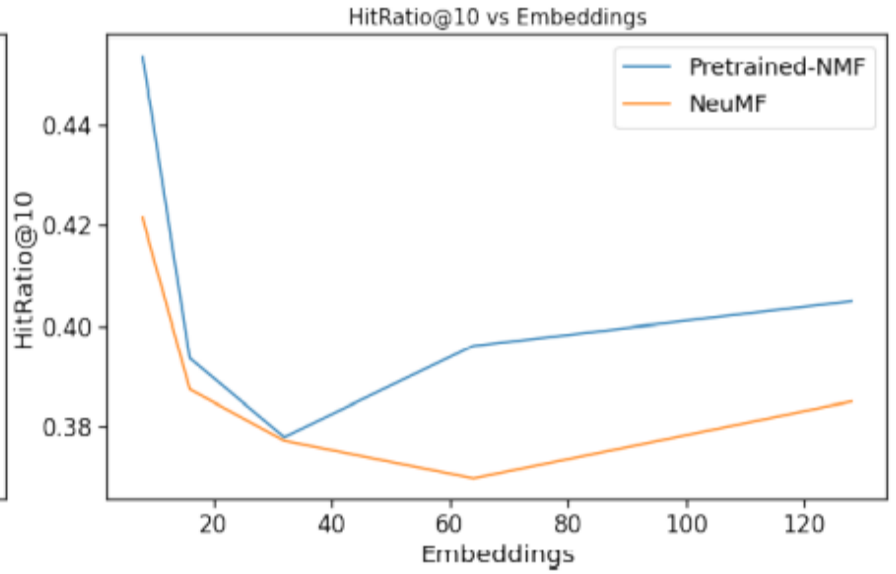
Performance Efficiency for pre-trained NeuMF model

Evaluation of NeuMF and pre-trained NeuMF w.r.t Embeddings

NeuMF			
Embeddings	**HitRate@10**	**NDCG@10**	**Time (s)**
8	0.4281	0.2350	477s
16	0.3874	0.2080	715s
32	0.3772	0.2038	930s
64	0.3697	0.2204	1,697s
128	0.4049	0.2184	1,132s

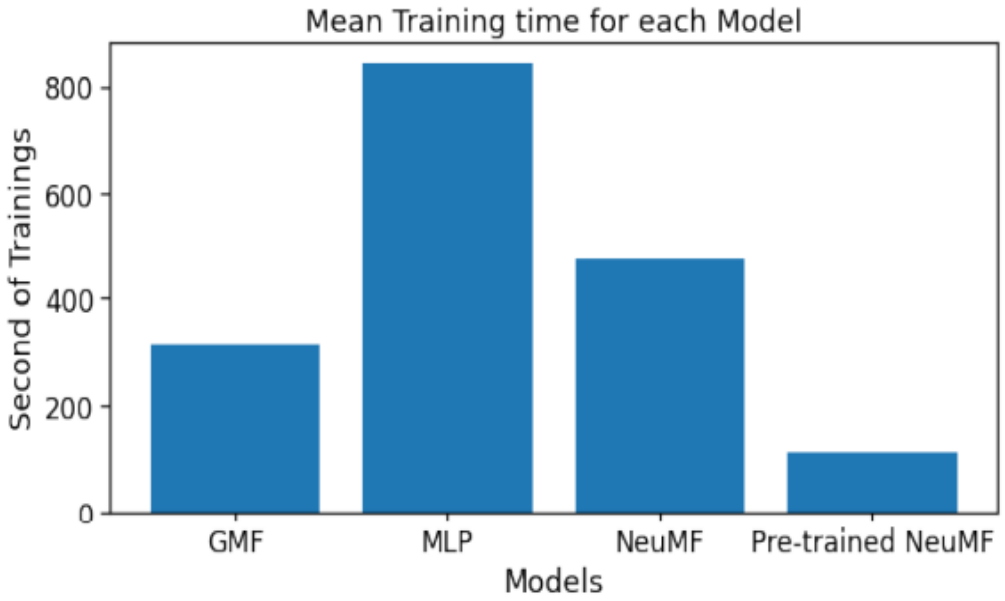
Pre-trained NeuMF			
Embeddings	**HitRate@10**	**NDCG@10**	**Time (s)**
8	0.4539	0.2531	122s
16	0.3936	0.2164	175s
32	0.3779	0.2161	196s
64	0.3961	0.2027	346s
128	0.4049	0.2316	260s





Training time for each model

Embeddings	GMF	MLP	NeuMF	pre-trained NeuMF
8	318.352	842.909	477.876	112.472
16	485.512	884.833	715.101	175.858
32	196.925	1.217.873	930.820	196.496
64	380.739	2.575.650	1.697.241	346.487
128	907.558	2.646.876	1.132.037	260.327



Thank you!