Movie Recommender System using Collaborative Filtering Techniques and Neural Collaborative Reasoning

Krishna Swaroop Pamidimukkala(kp956)
Rutgers University
kp956@scarletmail.rutgers.edu

Laxman Srikanth Sista (ls1274)
Rutgers University
ls1274@scarletmail.rutgers.edu

Abhinay Mannepalli (am2977)
Rutgers University
am2977@scarletmail.rutgers.edu

ABSTRACT

Every online video provider, from Netflix to Amazon, makes an effort to deliver appropriate movie suggestions. An effective movie recommendation system provides consumers with a selection of films that they might enjoy [1]. This is critical in order to increase client involvement and enthusiasm for the service. Users would be overwhelmed by choice without suggestions, which is a major issue with online movie libraries. Which technique to movie suggestions, on the other hand, is the most effective? This is the problem that we are attempting to tackle in our study. We developed Five traditional Collaborative Filtering based recommendation algorithms on different methodologies utilizing movie data given by GroupLens Research [2].

Because a user's future behavior may not be driven solely by its similarity to previous behaviors, but rather by the user's cognitive reasoning procedure about what to do next, recommendation, as a cognition rather than a perception task, necessitates not only pattern learning and matching, but also cognitive reasoning.

The authors suggested neural collaborative reasoning (NCR) [3], a framework to merge the capabilities of embedding learning with logical reasoning in the recommendation job, inspired by recent development on neural-symbolic machine learning.

KEYWORDS

Collaborative Filtering; Collaborative Reasoning; Recommender Systems; Cognitive Reasoning; Cognitive Intelligence; SVD; KNN; Alternative Least Squares (ALS); CoClustering; Neural Networks; Deep Learning

1 INTRODUCTION

With the expansion of data on the internet, surfing the web without any sort of external assistance appears to be impossible. Recommendation engines have played a big role in this. In its most basic form, a recommendation engine is an algorithm that suggests relevant products to users[1]. From E-commerce to social media, recommendation engines have aided in the navigation of billions of users who would otherwise be lost in a sea of information. They are extremely important in specific businesses, such as the internet video industry. Netflix, for example, offered a $1 million prize to anyone who could submit the best collaborative filtering algorithm for predicting user ratings.

In general, collaborative and content-based methods are the two types of recommendation systems [1]. Previous user interactions are significantly weighted in collaborative recommendation algorithms. They strive to understand how prior encounters between people and objects may influence future behavior as filtering engines. These user-item interactions are recorded and utilized to anticipate how users will engage in the future.

The content-based approach is the other general strategy taken by recommendation systems. Content-based approaches generate suggestions based on additional information about people or goods. The model seeks to develop a model based on extraneous information, such as the user's age or gender, or the film' category or director, in order to better explain user-item interactions.

These algorithms however only strive to capture the associative relevance patterns in data by learning user and item embeddings from data using models, such that a user embedding may be matched with relevant item embeddings using planned or learned similarity functions.

The Neural Collaborative Reasoning framework merge the power of embedding learning with logical reasoning in the recommendation task, based on recent advances in neural-symbolic machine learning. These representations record similarity patterns in data from perceptual viewpoints, and logic aids cognitive thinking for better recommendations. This modularized reasoning architecture learns logical operations like AND, OR, and NOT as neural modules.

Certain logical structures may appear in such a reasoning approach, such as $(a \vee b) \wedge \neg c \rightarrow v$, which suggests that if the user likes a or b but not c, he or she would most likely prefer v. An intelligent recommendation system, as a typical cognitive reasoning problem, should be able to uncover and utilize logical relationships in data in order to reason over the data for the prediction of future user actions. Researchers suggested NCR to attain this aim, which describes the recommendation issue as a differentiable (i.e., neural) logical reasoning problem based on wisdom of the crowd (i.e., collaborative). Each user's behavior record is interpreted as a logical implication rule, such as $(a \vee b) \wedge \neg c \rightarrow v$, which means that the user's behavior record is interpreted as given his or her past choices for a, b, and c, the user preferred item v. In this way, each user is aware of a portion of the entire logical space, allowing collaborative logical reasoning to be conducted based on the combined information of all users to estimate the preferences of new users.

In our project we will be using algorithms that fall within the collaborative category in order to investigate which approach is

Conference'17, Washington, DC, USA

DOI: 10.1145/nnnnnnn.nnnnnnn

superior. In addition, we have also implemented a Neural Collaborative Reasoning based on the collective information from all users to estimate the preferences for new users.

2 RELATED WORK

2.1 Singular Value Decomposition

Singular Value Decomposition is a collaborative recommendation engine technique for decomposing a matrix into three matrices which yield more information concerning the matrix data . It is defined as:

$A = U \sum VT$

where U is a m×m orthogonal matrix, $\sum$is a diagonal m×n matrix, V is a n×n orthogonal matrix.

2.2 K Nearest Neighbors

Another collaborative filtering technique, the KNN algorithm, is based on the simple concept that related objects are close to one other. Calculating cosine distances conveys this sense of similarity. The closer the distance between goods, the more likely they are to be similar. As a result, it can predict the label for these data based on cosine distances by locating the nearest training samples to a location. Despite its simplicity, KNN has proven to be effective in a variety of classification and regression applications.

2.3 Co-Clustering

Co-Clustering is a collaborative recommendation strategy that aims to cluster rows and columns of a matrix A at the same time given a matrix A. The purpose of coclustering, also known as bi-clustering, is to create bi-clusters, or a subset of rows that behave similarly across a subset of columns. With coclustering, two map functions are performed: rows are mapped to row cluster indexes, and columns are mapped to column cluster indexes. Co-clustering is a way of grouping two types of things together at the same time based on their pairwise interactions being similar. Optimization algorithms like k-means are used to run our co-clustering algorithm.

2.4 Alternative Least Squares

Alternative Least Squares (ALS) is a parallelized matrix factorization algorithm designed for large-scale collaborative filtering applications. ALS works well with sparse datasets and scales effectively. ALS works by alternately minimizing two loss functions. It first fixes the user matrix before running gradient descent with the item matrix; then it fixes the item matrix before running gradient descent with the user matrix. A cluster of machines runs gradient descent in parallel over numerous partitions of the training data.

2.5 Neural Collaborative Reasoning

Deep learning and neural network models have recently been developed, which has further expanded collaborative filtering approaches for recommendation. The applicable methods may be divided into two groups: similarity learning approaches and representation learning approaches. To calculate user-item matching scores, the similarity learning approach uses simple user/item representations (such as one-hot) and learns a complex matching function (such as a prediction network) [7, 18, 20], whereas the representation learning approach uses rich user/item representations and a simple matching function (e.g., inner product).

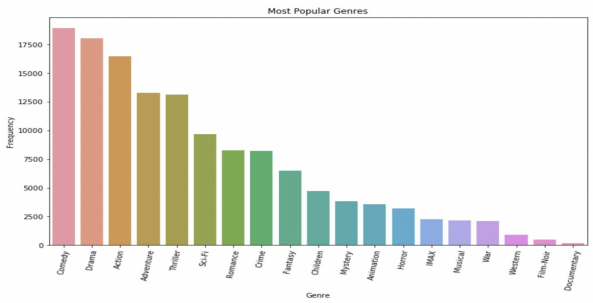
Despite the fact that several CF techniques for recommendation problems have been developed, most extant methods represent recommendation as a perceptual task based on similarity matching rather than a cognition job based on cognitive/logical reasoning. Users' future actions, on the other hand, may be motivated by a concrete reasoning mechanism about what to do next, rather than by a resemblance with their prior conduct. In numerous research domains, combining logical reasoning and neural networks has been proposed.

The goal of NCR as proposed by the authors is to embed logical expressions and logical propositional variables in a logical latent space, then use logical reasoning to deliver appropriate suggestions based on logical reasoning. As a result, NCR learns a vector representation for each propositional variable and logical phrase. During training, the logical latent space is also regularized to ensure that the neural modules exhibit the desired behavior.

3 ABOUT THE DATA

The dataset used for training and testing is Movielens100k part of Grouplens Research. There are 100k ratings/entries in the movie rating dataset and the ratings are ranging from 1 to 5. Some of the interesting finding about the data are no redundancies, missing values and no incorrect values, contains data authenticity, all the features are independent and positively skewed and none of the columns are normally distributed.

Exploratory Data Analysis-1



Comedy, Drama, and Action are the top 3 genres that are watched by viewers. These means that these three genres are more popular than the other genres.popular items get a lot of exposure while less popular ones are underrepresented in the recommendations.

Exploratory Data Analysis-2

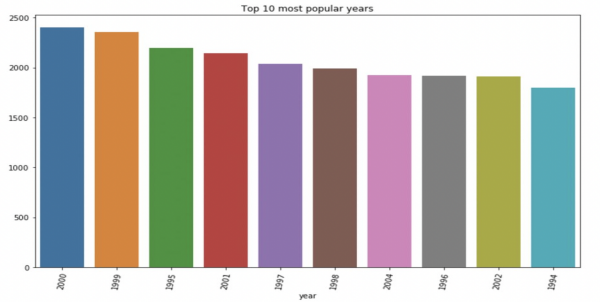
The films that are in 2000 year are more popular and are recommended more often than other films.

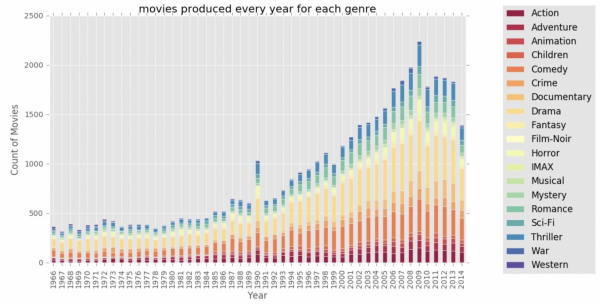
Exploratory Data Analysis-3

Number of movies have been increasing across the years without any gaps and number of movies per genres has also been increasing steadily until 2009.

Data Preprocessing for NCR

The ratings are transformed from the 1-5 scale to the 0-1 scale and the ratings equal to or higher than 4 are converted into a positive feedback (1), while ratings lower than 4 are converted into a negative feedback (0).Formalized the recommendation problem





into a logical reasoning problem using propositional logic. Since propositional variables are boolean variables,scale the ratings associated to the items in the 0-1 scale.Next group the ratings by user and ordered by timestamp. For each user the first 5 positive ratings are kept in training set in order to reduce the cold-start problem, the last positive rating is hold out for the test set, the second to the last positive rating is hold out for the validation set.Incase the user has less than 5 positive rating, all the ratings are kept in training set. If the user has only 6 positive ratings, 5 are kept in training set and the last one is used for test set. If the user has more than 6 positive ratings, the procedure explained before is used.For each user built their train, validation, and test logical expressions. These logical expressions are then used to train and test the NCR model.Logical expressions are built as follows:for each positive item in the user historical information, we select the 5 most recent items that come just before that specific item as the history set of the logical expression and then, built the logical expression using this template: (conjunction of items in the history set) $\rightarrow$ *selectedpositiveitem*.

4 PROBLEM FORMALIZATION

The primary purpose of personalized recommendations is to forecast a user's future behavior based on previous behavior. We start with implicit feedback from users, which means we only know if a user has engaged with an item but not if the user likes or dislikes it. Let's say a user's interaction history comprises r items $[v_1, v_2, .., v_r]$ and we're trying to figure out if an item v_x should be suggested to the user. To encode the interactions into a logic space, we need to create a function in first order logic. The dilemma of whether to suggest item v_x or not becomes a matter of assessing if the following Horn clause is True or False:

$$I(u, v_1) \wedge I(u, v_2) \wedge I(u, v_3) \wedge ... \wedge I(u, v_r) \rightarrow I(u, v_x)$$

Similarly for explicit feedback, given the set of items $[v_1, v_2, .., \neg v_r]$ where $\neg v_r$ is an item where the user has given a negative feedback, then the horn clause can be constructed as:

$$L(u, v_1) \wedge L(u, v_2) \wedge L(u, v_3) \wedge ... \wedge \neg L(u, v_r) \rightarrow L(u, v_x)$$

The left of the horn clause is the conjunction form of Historical Interactions and the right of the implication horn clause target interaction that needs to be predicted.

For the sake of simplicity, interaction of user u with item v_i is represented as a event vector $e_u^{v_i}$

5 THE PROPOSED MODEL

TRADITIONAL ALGORITHMS

We have implemented the traditional collaborative filtering methods to like SVD, KNN, Co-Clustering and ALS. Out of these SVD perfomed better as per the result. One thing which we observed were these were only trying to do pattern matching and not actually suggesting the next logical step so we went ahead and implemented NCR to overcome this.

NEURAL COLLABORATIVE REASONING

In order to derive the representation of the logical expression, an encoder is first utilized to link the user embedding with the item embedding, yielding the user-item pair's so-called event vector. This procedure is repeated for each user-item combination.

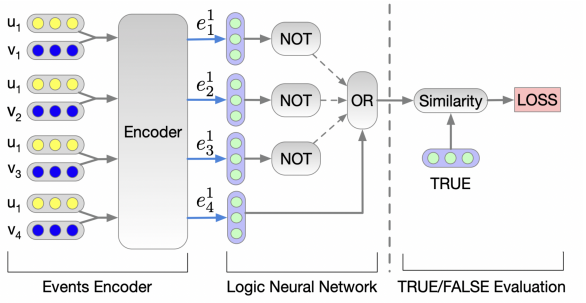
The calculated event vectors are then used to construct the logical statement.

- OR is a neural module that computes the logical OR operator of two input event vectors in the logical latent space in the architecture.
- NOT is similar to the OR module in that it performs the logical negation of the input event vector in the logical space. NOT is similar to the OR module in that it outputs a new event vector that is the result of the logical OR between the two input event vectors in the latent space.
- The TRUE logical constant is represented by a fixed latent vector called TRUE.
- Similarity is the cosine similarity between the event vector of the input logical expression and the TRUE vector;it is utilized as an anchor vector to learn the user and item embeddings. If the output is high, the item v can be suggested to user u; however, if the output is low, the item v should not be recommended to user u. The Similarity between the Final event embedding vector and TRUE vector is calculated as follows:

$$Sim(\mathbf{EXP},\mathbf{T}) = \frac{\mathbf{EXP}.\mathbf{T}}{\|\mathbf{EXP}\|.\|\mathbf{T}\|}$$

5.1 Pairwise Loss function and Training

The pair-wise learning approach is used by NCR. In other words, we construct a negative logical expression for each logical expression in the training set. Instead of the target item on the right side of the implication, the negative logical expression has a randomly picked item from the collection of objects that the user never interacted with. We urge the original (positive) logical expression to be as close to the TRUE vector in the latent space as possible during training, while we force the negative expression to be as far away from the



TRUE vector in the latent space as feasible. This is accomplished by using gradient descent to minimize the following loss function:

$$\mathcal{L}_{ncr} = -\sum_{u\in\mathcal{U}}\sum_{v_i\in\mathcal{V}_u^+}\sum_{v_j\in\mathcal{V}_u^-} \ln\sigma(s_{uij}) + \lambda_\Theta\|\Theta\|_2^2 \tag{1}$$

where $s_{uij} = \alpha\cdot(s_{ui}^+ - s_{uj}^-)$ $s_{ui}^+ and s_{uj}^-$ are the truth evaluation results which are calculated using expression for the observed ground-truth interaction (C_{ui}^+) and the expression for the negative sampled interaction (C_{ui}^-) respectively using the expressions:

$$\begin{aligned} s_{ui}^+ &= Sim\big(\text{LNN}(\mathbf{e}_u^{v_{i-1}}, \mathbf{e}_u^{v_{i-2}}, \cdots, \mathbf{e}_u^{v_{i-r}}, \mathbf{e}_u^{v_i}),\big) \\ s_{uj}^- &= Sim\big(\text{LNN}(\mathbf{e}_u^{v_{i-1}}, \mathbf{e}_u^{v_{i-2}}, \cdots, \mathbf{e}_u^{v_{i-r}}, \mathbf{e}_u^{v_j}),\big) \end{aligned} \tag{2}$$

s_{uij} to represent the difference between these two expressions and use an optimization technique to maximize it, where is an amplification factor to magnify the difference (in our case, $\alpha = 10$)

5.2 Logical Modules and Regularizers

The NCR model adds an extra regularization term to the loss function mentioned above during training. This phrase refers to the process of applying logical regularization to the logical latent space. These regularizers are employed to compel the neural modules to function appropriately, specifically to allow them to work as expected, because it is not certain that they would act like the logical operators. For example, we want to make sure that the OR neural module delivers an event vector in the logical latent space that is the logical OR of the two input event vectors. Integrate the logic regularizer together with the pairwise learning loss to get the final loss function as follows:

$$\mathcal{L} = \mathcal{L}_{ncr} + \lambda_r\mathcal{L}_{logicReg} \tag{3}$$

In the table below, you'll find the logical regularizers.

6 EXPERIMENTS

We began our experiment by first dividing the dataset into a training set and testing set. Then we used the training set data to train our algorithms. These training algorithms were then used to predict movies for users in the testing set.

Evaluation metrics for Traditional Algorithms (SVD, KNN, ALS and CoClustering):

- Root Mean Square Error (RMSE)
- Mean Absolute Error (MAE)

Evaluation metrics for NCR:

- Pairwise Loss Function

	Law	Equation	Logical Regularizer r_i
NOT	Negation	$\neg T = F$	$r_1 = \frac{1}{\lvert\mathcal{X}\rvert}\sum_{\mathbf{x}\in\mathcal{X}} 1 + Sim(\text{NOT}(\mathbf{x}), \mathbf{x})$
	Double Negation	$\neg(\neg x) = x$	$r_2 = \frac{1}{\lvert\mathcal{X}\rvert}\sum_{\mathbf{x}\in\mathcal{X}} 1 - Sim(\text{NOT}(\text{NOT}(\mathbf{x})), \mathbf{x})$
AND	Identity	$x \wedge T = x$	$r_3 = \frac{1}{\lvert\mathcal{X}\rvert}\sum_{\mathbf{x}\in\mathcal{X}} 1 - Sim(\text{AND}(\mathbf{x}, T), \mathbf{x})$
	Annihilator	$x \wedge F = F$	$r_4 = \frac{1}{\lvert\mathcal{X}\rvert}\sum_{\mathbf{x}\in\mathcal{X}} 1 - Sim(\text{AND}(\mathbf{x}, F), F)$
	Idempotence	$x \wedge x = x$	$r_5 = \frac{1}{\lvert\mathcal{X}\rvert}\sum_{\mathbf{x}\in\mathcal{X}} 1 - Sim(\text{AND}(\mathbf{x}, \mathbf{x}), \mathbf{x})$
	Complementation	$x \wedge \neg x = F$	$r_6 = \frac{1}{\lvert\mathcal{X}\rvert}\sum_{\mathbf{x}\in\mathcal{X}} 1 - Sim(\text{AND}(\mathbf{x}, \text{NOT}(\mathbf{x})), F)$
OR	Identity	$x \vee F = x$	$r_7 = \frac{1}{\lvert\mathcal{X}\rvert}\sum_{\mathbf{x}\in\mathcal{X}} 1 - Sim(\text{OR}(\mathbf{x}, F), \mathbf{x})$
	Annihilator	$x \vee T = T$	$r_8 = \frac{1}{\lvert\mathcal{X}\rvert}\sum_{\mathbf{x}\in\mathcal{X}} 1 - Sim(\text{OR}(\mathbf{x}, T), T)$
	Idempotence	$x \vee x = x$	$r_9 = \frac{1}{\lvert\mathcal{X}\rvert}\sum_{\mathbf{x}\in\mathcal{X}} 1 - Sim(\text{OR}(\mathbf{x}, \mathbf{x}), \mathbf{x})$
	Complementation	$x \vee \neg x = T$	$r_{10} = \frac{1}{\lvert\mathcal{X}\rvert}\sum_{\mathbf{x}\in\mathcal{X}} 1 - Sim(\text{OR}(\mathbf{x}, \text{NOT}(\mathbf{x})), T)$
AND/OR	Associativity	$x \vee (y \vee z) = (x \vee y) \vee z$ $x \wedge (y \wedge z) = (x \wedge y) \wedge z$	Random Shuffling of Logic Variables
	commutativity	$x \vee y = y \vee x$ $x \wedge y = y \wedge x$	

Relevance

- Precision
- Recall
- nDCG
- F-Measure

We have performed 3 major experiments in order to understand the performance of the NCR framework in much more detail:

- **Experiment 1** : Performance of NCR on varying number of premises
- **Experiment 2** : Performance of NCR on different embedding sizes
- **Experiment 3** : Performance of NCR on varying the L2 regularization parameter

EXPERIMENT 1 : PERFORMANCE OF NCR ON VARYING NUMBER OF PREMISES

We have performed a few experiments by filtering logical statements based on the number of premises threshold . Before the model is trained, any logical expressions having a number of premises equal to or less than premise threshold are deleted from the dataset.

We observe from the results that the the when the logical expressions are limited to 2 or more premises, we observe an improvement in across all the metrics i.e nDCG, Precision, Recall, F-measure for K=10 but do not observe the same for K=5 as compared to the baseline model.

However, all the metrics seem have less value as compared to the baseline model when the number of premises have been restricted to exactly 5 premises.(Refer to Table 1 for results and Figure 5 and 6 for loss plots)

EXPERIMENT 2 : PERFORMANCE OF NCR ON DIFFERENT EMBEDDING SIZES

We have also performed size of users, item, and event embeddings. The greater the size of the embeddings vector, the richness/quality of the embeddings improve. Embeddings with much more information will help the model perform better the baseline.

Premises in the Logical Expression	NDCG @5	NDCG @10	Precision @5	Precision @10	Recall @5	Recall @10	Fmeasure @5	Fmeasure @10	Loss
Baseline	0.37	0.427	0.495	0.695	0.526	0.702	0.51	0.70	13.4
More than 2	0.37	0.428	0.494	0.699	0.525	0.703	0.509	0.701	14.1
Exactly 5	0.37	0.423	0.493	0.604	0.526	0.690	0.509	0.644	13.3
No Double Negations	0.362	0.417	0.496	0.594	0.517	0.689	0.506	0.638	16.3

Figure 1: Performance of NCR on varying number of premises

Embedding Size	nDCG @5	nDCG @10	Precision @5	Precision @10	Recall @5	Recall @10	Fmeasure @5	Fmeasure @10	Loss
Baseline (32)	0.35	0.406	0.495	0.592	0.501	0.674	0.50	0.63	16.8
64	0.37	0.427	0.497	0.674	0.526	0.702	0.511	0.687	15.5
128	0.37	0.427	0.506	0.648	0.535	0.696	0.52	0.67	13.2
256	0.37	0.430	0.496	0.594	0.534	0.706	0.514	0.645	14.5

Figure 2: Performance of NCR on different embedding sizes

	nDCG @5	NDCG @10	Precision @5	Precision @10	Recall @5	Recall @10	Fmeasure @5	Fmeasure @10	Loss
L2 Decrease	0.365	0.421	0.495	0.495	0.513	0.688	0.503	0.576	14.3
Full Model	0.37	0.427	0.497	0.494	0.526	0.702	0.511	0.58	15.5
L2 Increase	0.361	0.416	0.506	0.504	0.512	0.685	0.509	0.581	17.0

Figure 3: Performance of NCR on varying the L2 regularization parameter

Model	Precision	Recall	F-Measure	NDCG	MAE	RMSE
SVD	0.683	0.683	0.683	0.95	0.684	0.872
KNN	0.677	0.677	0.677	0.735	0.705	0.894
Co Clustering	0.676	0.676	0.676	0.686	0.718	0.916
ALS	0.651	0.651	0.651	0.712	0.681	0.872

Figure 4: Performance of Traditional Algorithms

The embedding vector size is set to 64 in the baseline model. As we decrease the size of the embeddings, all the accuracy metrics like nDCG, Precision, Recall have dropped as compared to the baseline.

On the other hand, cases when we have increased the sizes of embeddings to 128 and 256, all the accuracy metrics like nDCG, Precision, Recall have improved for both K=5 and K=10. (Refer to Table 2 for results)

EXPERIMENT 3 : PERFORMANCE OF NCR ON VARYING THE L2 REGULARIZATION PARAMETER

Weights are pushed toward zero by L2 regularization, although they are not precisely zero. At each iteration, L2 regularization operates as a force, removing a tiny proportion of weights. As a result, weights will never equal zero.

NCR uses the ℓ_2-regularization and dropout to prevent overfitting. The weight of ℓ_2-regularization λ_Θ is set between 1×10^{-6} to 1×10^{-4}

We observe some minor fluctuations in the results as we increase and decrease the ℓ_2-regularization λ_Θ value. The highest accuracy is observed for the baseline configuration.(Refer to Table 3 for results and Figure 7 and 8 for loss plots)

PERFORMANCE OF TRADITIONAL ALGORITHM - SVD, KNN, COCLUSTERINF AND ALS

From the above table (Refer to Table 4) we can observe that SVD performed better in comparison with others. In general SVD is better for explicit data and ALS is better for implicit data. Our dataset is explicit.

7 CONCLUSIONS AND FUTURE WORK

We have implemented various traditional algorithms in order to understand the quality of recommendations provided. Based on the shortcomings of the recommendations and limitations of these traditional algorithms to move beyond just pattern recognition, we have implemented NCR framework.

Using the Neural Collaborative Reasoning (NCR) framework for customized recommendation, which treats recommendation as a reasoning process by merging logical structures and neural networks. Experiments have shown that these technique improves ranking performance significantly given the quality of recommendations. We ran further tests to investigate our model's behavior in various scenarios in order to better understand why it performs well. The findings suggest that logical regularization is beneficial to recommendation performance. NCR framework has only employed user interaction information for collaborative reasoning in this study, but it would be interesting to explore contextual and multimodal information for reasoning in the future.

ACKNOWLEDGEMENT

We would like to thank Professor Yongfeng Zhang and his team for their support and guidance throughout the course and the final project. We believe his inputs have proved to be extremely valuable to us.

REFERENCES

[1] Gediminas Adomavicius, Bamshad Mobasher, Francesco Ricci, and Alex Tuzhilin. 2011. Context-Aware Recommender Systems. Recommender systems handbook (2011), 217–253.

[2] "MovieLens." GroupLens. 2021.

[3] Hanxiong Chen, Shaoyun Shi, Yunqi Li, and Yongfeng Zhang. Neural collaborative reasoning. In Proceedings of the Web Conference 2021, pp. 1516–1527, 2021.

[4] Yoshua Bengio. 2019. From System 1 Deep Learning to System 2 Deep Learning. In NeurIPS'2019.

[5] Tarek R Besold, Artur d'Avila Garcez, Sebastian Bader, Howard Bowman, Pedro Domingos, Pascal Hitzler, Kai-Uwe Kühnberger, Luis C Lamb, Daniel Lowd, Priscila Machado Vieira Lima, et al. 2017. Neural-symbolic learning and reasoning: A survey and interpretation. arXiv preprint arXiv:1711.03902 (2017).

[6] Xu Chen, Hongteng Xu, Yongfeng Zhang, Jiaxi Tang, Yixin Cao, Zheng Qin, and Hongyuan Zha. 2018. Sequential recommendation with user memory networks. In WSDM. 108–116.

[7] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide deep learning for recommender systems. In DLRS: Proceedings of the 1st RecSys workshop on deep learning for recommender systems. 7–10.

[8] Maurizio Ferrari Dacrema, Simone Boglio, Paolo Cremonesi, and Dietmar Jannach. 2021. A troubling analysis of reproducibility and progress in recommender systems research. ACM Transactions on Information Systems (TOIS) (2021).

[9] Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. 2019. Are we really making much progress? A worrying analysis of recent neural recommendation approaches. In RecSys. 101–109.

[10] Wang-Zhou Dai, Qiuling Xu, Yang Yu, and Zhi-Hua Zhou. 2019. Bridging machine learning and logical reasoning by abductive learning. In NeurIPS. 2815–2826.

[11] Honghua Dong, Jiayuan Mao, Tian Lin, Chong Wang, Lihong Li, and Denny Zhou. 2019. Neural logic machines. ICLR. [12] Michael D Ekstrand, John T Riedl, Joseph A Konstan, et al. 2011. Collaborative filtering recommender systems. Foundations and Trends® in Human–Computer Interaction 4, 2 (2011), 81–173.

[13] Thomas Elsken, Jan Hendrik Metzen, Frank Hutter, et al. 2019. Neural architecture search: A survey. Journal of Machine Learning Research 20, 55 (2019), 1–21.

[14] Maurizio Ferrari Dacrema, Federico Parroni, Paolo Cremonesi, and Dietmar Jannach. 2020. Critically Examining the Claimed Value of Convolutions over User-Item Embedding Maps for Recommender Systems. In CIKM. 355–363. [15] Artur S d'Avila Garcez, Krysia B Broda, and Dov M Gabbay. 2012. Neural-symbolic learning systems: foundations and applications. Springer Sci. Bus. Media.

[16] Sumit Gulwani, Oleksandr Polozov, Rishabh Singh, et al. 2017. Program synthesis. Foundations and Trends® in Programming Languages 4, 1-2 (2017), 1–119.

[17] F Maxwell Harper and Joseph A Konstan. 2016. The movielens datasets: History and context. Acm TIST 5, 4 (2016), 19.

[18] Ruining He, Wang-Cheng Kang, and Julian McAuley. 2017. Translation-based recommendation. In RecSys. 161–169.

[19] Ruining He and Julian McAuley. 2016. VBPR: Visual Bayesian Personalized Ranking from Implicit Feedback. In AAAI.

[20] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In WWW. 173–182.

[21] https://github.com/bmxitalia/NCRProject.git

[22] https://github.com/rutgerswiselab/NCR.git

[23] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and D Tikk. 2016. Session-based recommendations with recurrent neural networks. In ICLR.

8 NCR MODELS-LOSS AND NDCG CURVES

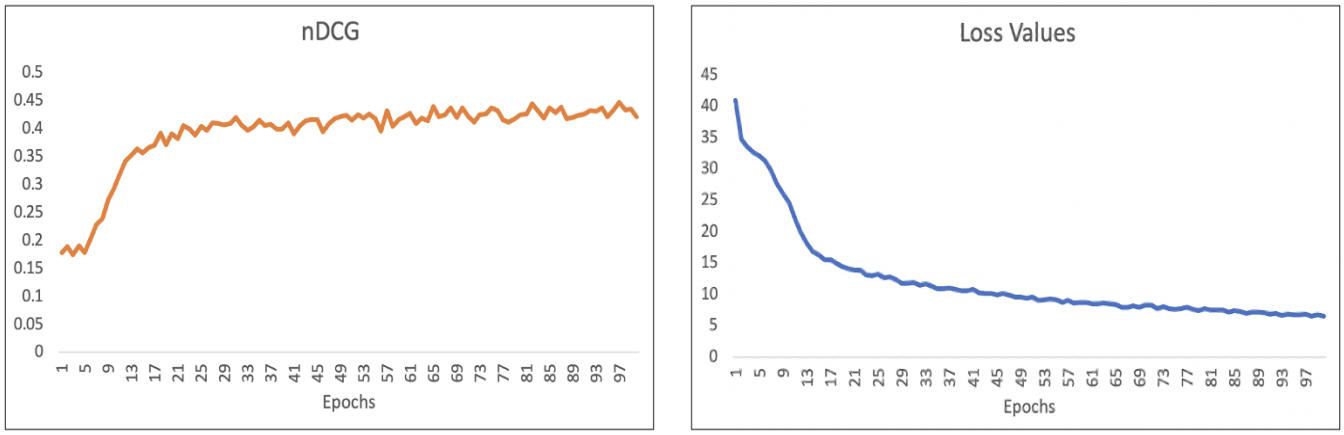


Figure 5: Baseline NCR Model Performance

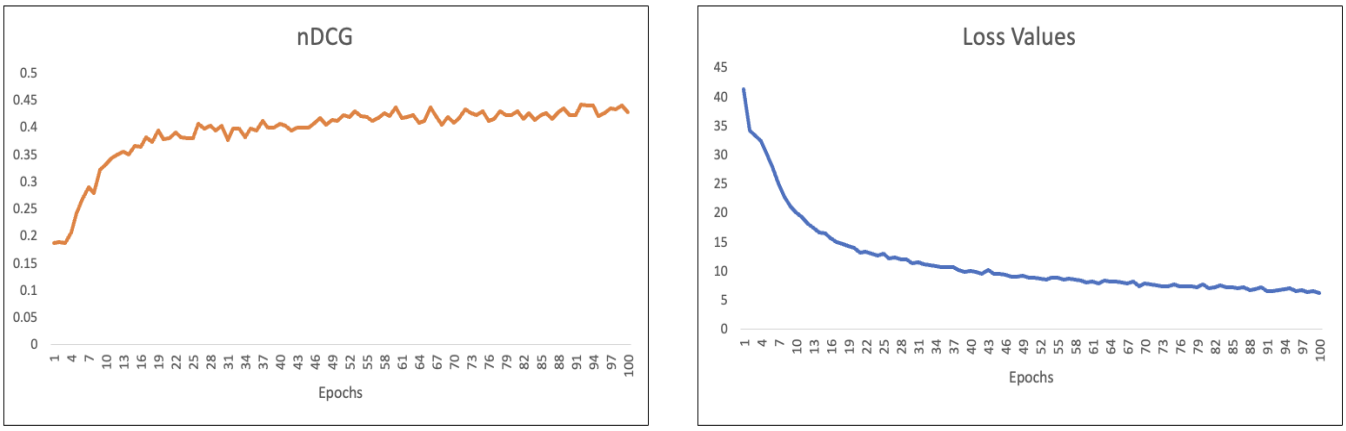


Figure 6: NCR Model Performance with 2 or more Premises

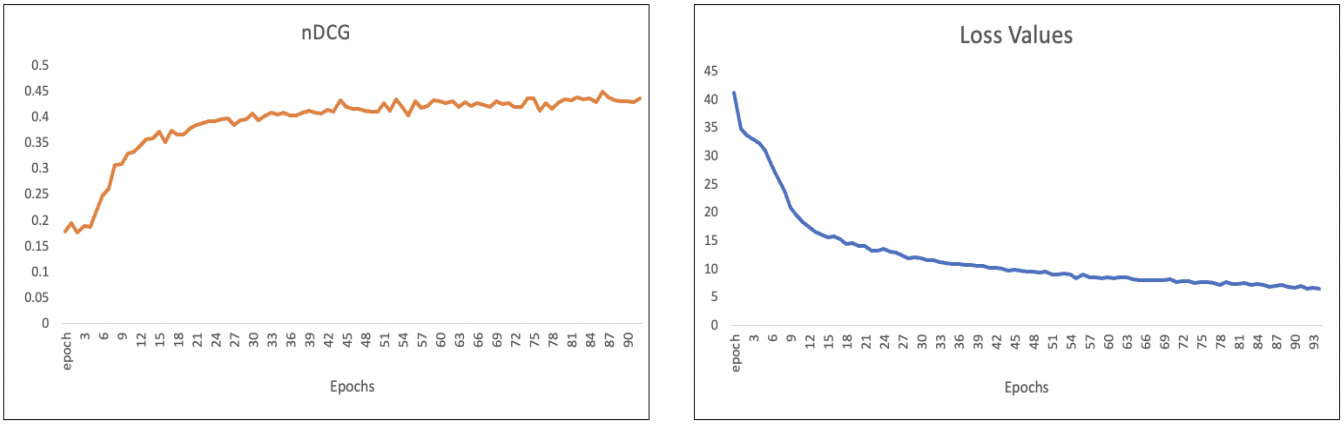


Figure 7: Decrease of L2 Regularization Value

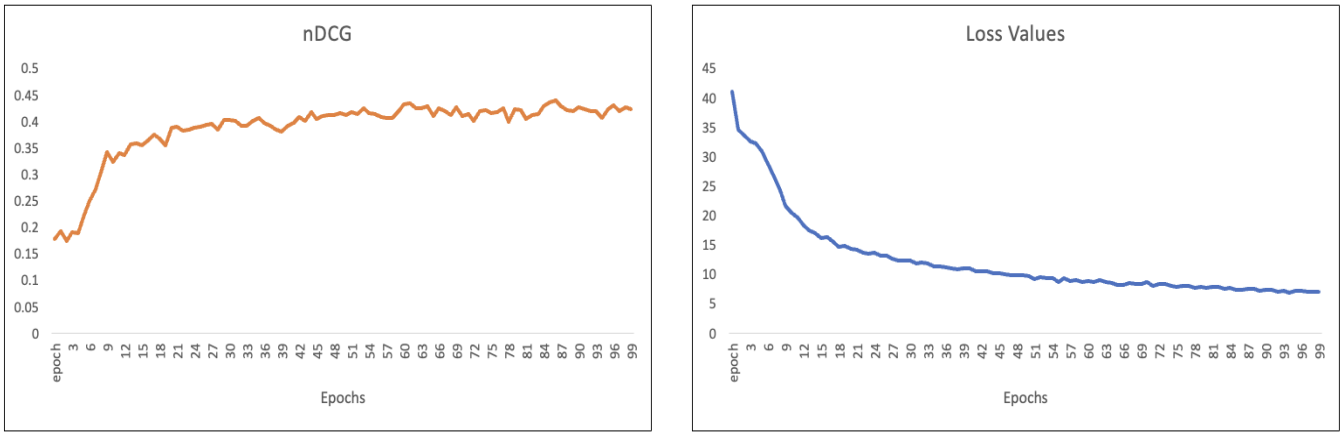


Figure 8: increase of L2 Regularization Value