Received November 26, 2021, accepted March 5, 2022, date of publication March 14, 2022, date of current version March 24, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3159646

Evaluating Top-N Recommendations Using Ranked Error Approach: An Empirical Analysis

SOFIA AFTAB AND HERI RAMAMPIARO
Department of Computer Science (IDI), Norwegian University of Science and Technology (NTNU), 7491 Trondheim, Norway

Corresponding author: Sofia Aftab (sofia.aftab@ntnu.no)

This work was supported by the IDI Department of NTNU.

ABSTRACT Evaluation metrics or measures are necessary tools for evaluating and choosing the most appropriate modeling approaches within recommender systems. However, evaluation measures can sometimes fall short when evaluating recommendation lists that best match users' top preferences. A possible reason for this shortcoming is that most measures mainly focus on the list-wise performance of the recommendations and generally do not consider the item-wise performance. As a result, a recommender system might apply a weak or less accurate modeling approach instead of the best one. To address these challenges, we propose a new evaluation measure that incorporates the rank order of a prediction list with an error-based metric to make it more powerful and discriminative and thus more suited for top-N recommendations. The main goal of the proposed metric is to provide recommender systems, developers and researchers an even better tool, which enables them to choose the best modeling approach possible, and hence maximizing the quality of top-N recommendations. To evaluate the proposed metric and compare its general properties against existing metrics, we perform extensive experiments with detailed empirical analysis. Our experiments and the analysis show the usefulness, effectiveness and feasibility of the new metric.

INDEX TERMS Recommender system, evaluation metric, ranking measures, error-based metrics, neural network, user preferences.

I. INTRODUCTION

With the proliferation and advancement of recommender systems in recent years, the evaluation of recommendation approaches has become a crucial but a challenging task. Essentially, a recommender system is expected to perform three important steps: (1) understand the user's needs, interests and preferences, (2) identify the items that satisfy the user with respect to (1), (3) rank the suggested items according to user's preference structure (true preference order of items). To be able to assess how well a recommender system performs these tasks, an evaluation measure is a natural tool [1]. Hence, it is paramount that the evaluation measure is able to exactly evaluate the effectiveness of the recommendation while taking into account the aforementioned tasks. For this to be possible, the evaluation of a recommender system needs to determine *what* specifically contributes to good recommendations and *how* that is measured.

In early researches, recommender systems were considered as rating predictors [2], [3], with an intuition that correct rating prediction can provide valuable recommendations by suggesting the items with the highest predicted scores. However, later work proposed that the evaluation of recommendation methods using error-based metrics provides a weak proxy for true user satisfaction in real applications [2]–[4]. One of the arguments in support of this approach is that recommender systems in production typically deal with a shortlist of recommendations and the predicted ratings are often not visible to the user [2]. Generating a shortlist of N recommendations per user instead of focusing on accurately predicting the rating values that users would assign is commonly known as top-N recommendation [4]. The goal of this task is to provide a list of the most preferred items to the user. Additionally, evaluation meant for rating prediction gives equal importance to all items. As opposed to this, top-N recommendation inherently concerns the top-ranked items that a user may browse in the recommended list only. For these reasons, much of the attention has been shifted from rating prediction problems to top-N recommendation problems, i.e., ranking measures.

It is important to note that predictive accuracy has been ignored and replaced by predictive order by shifting from rating-based to ranking-based recommender systems. However, accuracy has generally been considered a prerequisite

The associate editor coordinating the review of this manuscript and approving it for publication was Fan Zhang.

for recommendation to be useful [5]. Therefore, we may say that the quality of rank-based measures can be degraded if the following two scenarios occur related to low predictive accuracy: (1) an item is less relevant to the user but appears early in the recommended list due to a high predicted score (up-graded relevancy); or (2) an item is highly relevant to the user but the recommendation algorithm does not consider it relevant for the user, due to low predicted score (degraded relevancy (see Figure 1). Such a conflict may also lead to a situation in which the user's favorite item is completely removed from the recommended list due to a cutoff (i.e., top-N items are chosen). This might, in turn, result in a person losing interest in shopping because their least favorite items continuously appear in top-N recommendations.

Commonly used measures in recommender systems, like Mean average precision (MAP) and Normalized discounted cumulative gain (NDCG), do not deal with the above-mentioned phenomenon and only pay attention to maximizing the precision (accuracy) or gain (utility) at the list level. For example, consider the following two lists: [5, 5, 3, 3, 3] and [3, 3, 5, 5, 5]. Applying a cutoff at 3, MAP-related measures cannot differentiate these two lists. Both lists are accurate, but the first one is more appropriate in the context of top-N recommendations. Further, consider the following two lists: [3, 3, 3, 2] and [5, 5, 4, 4, 3]. Similar to MAP, NDCG-related measures cannot differentiate these two lists. Both measures bring the most relevant items earlier within each list, but the second list is more appropriate and comprehensive in the context of top-N recommendations, since it provides more value to the user in terms of relevancy. We conclude that existing ranking measures do not capture the rating migration pattern between a relevance list and a recommended list, and therefore, top-N recommendations may not be very useful for the end-user.

In addition to the aforementioned challenges of rank-based measures, we would also like to highlight the fact that in contrast to the information retrieval (IR) domain, there are some applications in the recommendation domain where visibility of predicted score, in addition to rankings, adds more value to the end-user. For example, Amazon retail services provide rating details to enhance users' trust and satisfaction levels. Therefore, metric, which can differentiate between low and high score values is desirable for top-N recommendations. Moreover, any recommendations to the user may not be considered trustworthy unless the user is able to verify or quantify them. This trust may be weakened if incorrect or low predicted scores are assigned to top-ranked items or the users do not find their favorite items due to their absolute rating being below a cutoff value. This makes the accuracy of predicted scores a vital part of recommender systems. As a bonus, incorporating a predictive score can also be used to confirm the predictive power and performance of a recommendation model, which are all critical for recommender systems in production. It can also uncover issues, such as biases and over/under-fitting of the model.

Finally, there has been a consensus in the literature that accuracy as a broad concept remains a fundamental requirement for a recommendation to make sense [5], aside from the increasing interest for additional complementary recommendation properties (such as diversity and novelty) [2], [3]. For a given user, we might say that a particular suggestion is "correct" if that user likes or assign a high rating to the recommended item. With this goal in mind, we propose to bridge the gap between accuracy-based and ranking-based metrics. We argue that ranking of items can provide more value to the user if accuracy is also enforced through rank in top-N recommendations.

All the above-mentioned challenges were the main reasons for us to introduce a novel evaluation measure. We call this measure *Standard Discounted Cumulative Squared Error (SDCSE)*. SDCSE aims at capturing the utility of items through a ranking approach along with the accuracy of recommendations. It incorporates a list-wise comparison, as well as an item-wise comparison to improve the performance and discriminative power of the metric, which suits well with top-N recommendations. This is done by exploiting a ranked list with respect to predictive accuracy for each item, thus enabling the recommender system to enforce both the accuracy and utility at the same time.

The remainder of the paper is organized as follows. Section II gives an overview of existing methods and discusses their relation to our approach. It also presents the background of limitations of existing measures and thus the need for a new metric. Section III describes the standard discounted cumulative squared error approach in detail. Section IV outlines the experimental settings, including the datasets used and the comparative analysis of the proposed metric with existing measures. This section also provides insights into the modeling algorithms' contributions towards the relevancy score. Section V presents the results for methods discussed in Section IV, Finally, Section VI concludes the paper and outlines possible future research topics.

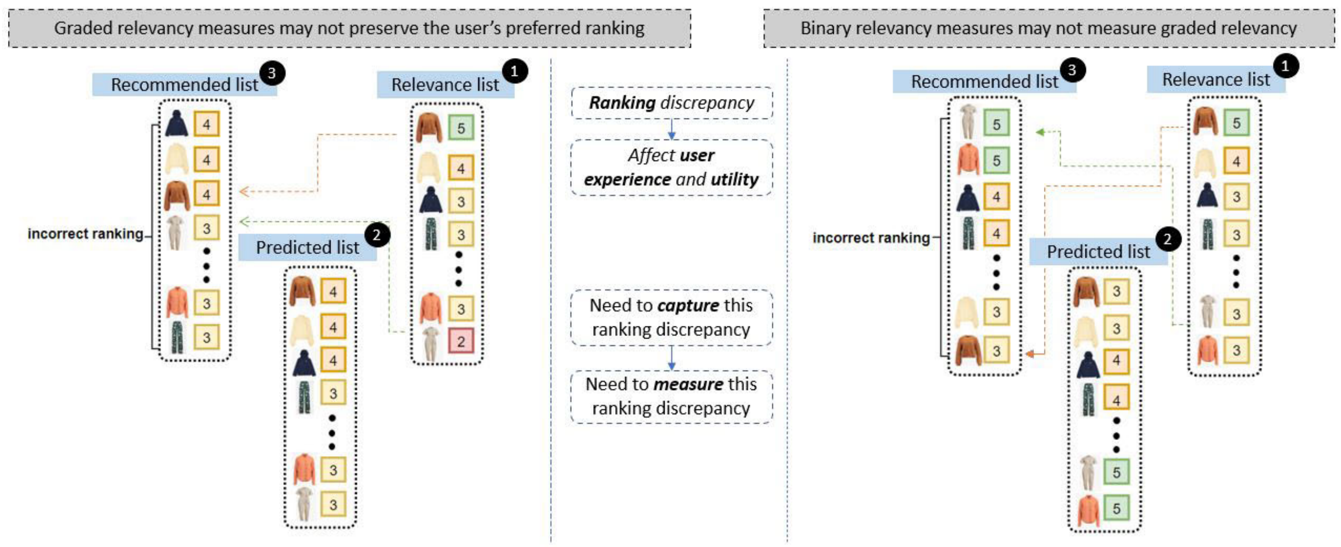
II. BACKGROUND AND RELATED WORK

A. RELATED WORK

Traditionally, *accuracy-based* metrics have been the most common methods for offline evaluation [2]. There are number of methods that fall into this category [1], including Error metrics [2], Precision and Recall metrics, ROC curves, and Ranking Score [6]. Our proposed method falls under the umbrella of Error metrics and Raking scores. For this reason, we only discuss these measures.

1) ERROR METRICS

Error-based metrics are the most widely-used methods for measuring predictive accuracy. For example, *Mean Absolute Error* [2] evaluates the difference between actual and predicted ratings. *Root Mean Squared Error* is another metric that gives more weight to the larger errors [6]. There are some other variations of these metrics like *Average MAE*, *Average*


Graded relevancy measures may not preserve the user's preferred ranking
Recommended list 3
Relevance list 1
Predicted list 2
incorrect ranking
Ranking discrepancy
Affect user experience and utility
Need to capture this ranking discrepancy
Need to measure this ranking discrepancy
Binary relevancy measures may not measure graded relevancy
Recommended list 3
Relevance list 1
Predicted list 2
incorrect ranking

- **Relevancy:** Relevancy for binary measures ranging from 3-5 (1-2 are irrelevant) while multi graded measures incorporate all ratings levels
- **Relevance list**: Actual/true ratings list generated by user (true user's preference structure/ranking)
- **Predicted list**: Predicted ratings list generated by system/model
- **Recommended list**: Sorted-predicted list presented to end user by the ranking function (ranked list)
- **Dotted-arrows:** Represents up-graded and down-graded relevancy

FIGURE 1. The right-hand side of the diagram shows a scenario where the evaluation metric cannot measure graded relevancy; whereas the left-hand side of the diagram depicts a scenario, where the evaluation metric does not consider the accuracy of the model, and hence the application of a weak model might not be identified by the evaluation metric, resulting in perfect score for a less relevant list.

RMSE and *Mean Squared Error*. However, these metrics treat all items equally. Hence, they may not be suitable for a list-wise evaluation approach for the recommendation.

2) RANKING METRICS

Herlocker *et al.* [2] argued that ranking measures are suitable for list-wise evaluation. These measures assign higher priority to more relevant items than the less relevant ones. For example, the R score metric [6] and NDCG score [7] give more weight to the items at the top of the list than those at the bottom of the list. However, these metrics ignore the predicted relevancy scores and use predicted orders instead. Therefore, these evaluation measures might not be able to differentiate between strong and weak models.

B. LIMITATIONS OF EXISTING RECOMMENDATION METRICS

There is a variety of recommendation methods and techniques used for filtering items. To choose the best among these methods and techniques, one of the primary decision factors is the quality of recommendations, which refers to how well the recommendation meets the user's needs and how much the recommended items are relevant to the user. This section explores existing widely-used measures and discusses why these are not always sufficient to fully evaluate a recommender system.

Formally, a rating/relevance from user u to the item j can be written as $r(u, j)$ and the relevance vector for each user can be represented as *relvec*, if the relevancy is greater than a certain threshold. Otherwise, it is *nrelvec*, (rating 4 could be a threshold in rating range from 1 to 5). Similarly, the predicted vector *predvec* consists of the predicted scores for N items. Each *relvec* and each *predvec* are within $[1, 5]$ and $[0, 1]$ intervals for explicit and implicit feedback, respectively. N denotes top-N items in the list and i is the index of item j.

Precision (P) defines how well a method places relevant items in top-N recommendations.

$$P_u@N = \frac{relvec \cap predvec}{n} \tag{1}$$

Recall (R) calculates the number of relevant items for a given user, that are included in the top-N recommendation as compared to the total number of relevant items.

$$R_u@N = \frac{relvec \cap predvec}{relvec} \tag{2}$$

F-Measure (F1) is the harmonic mean of P and R, which is used to adjust the P and R contribution.

$$F1_u@N = \frac{2.P_u@N.R_u@N}{P_u@N + R_u@N} \tag{3}$$

Average Precision (AP) computes the average of precision of all precision values at all positions where a relevant item is found. Finally, *MAP* is calculated by taking mean over all users in recommended system.

$$AP@N = \frac{1}{relvec}\sum_{i=1}^{N}(relvec_i \in predvec)P_u@N \quad (4)$$

Another closely related metric to AP is B$pref$. It is designed to be more robust than AP to incomplete relevancy [8].

$$Bpref_u@N = \frac{1}{\sum_{i=1}^{N}}(predvec_i \in relvec) \times (1 - \frac{min(predvec \cap nrelvev, relvec)}{min(nrelvev, relvec)}) \quad (5)$$

Another similar metric to MAP is Inferred Average Precision (InfAP) which generates the same score as MAP, when the relevance values are complete. However, in case of incomplete relevancy, it can also be considered as a statistical estimate of MAP [9].

$$InfAP_u@N = \frac{1}{\sum_{i=1}^{N}}(predvec_i \in relvec)(E[P_u@i]) \quad (6)$$

Here EP@i is defined as below and η is very small constant

$$E[P_u@i] = \frac{1}{i} + \frac{i-1}{i} \times (\frac{predvec_i \in relvec + \eta}{predvec_i \in relvec + predvec_i \in nrelvev + 2\eta}) \quad (7)$$

(MAP)@N [10] assigns more weight to the mistakes at the top of the list than at the bottom of the list. This metrics only determines the relevancy and non-relevancy of items and does not deal with the fine-grained impact of relevancy. For example, less relevant items are considered the same as the more relevant ones.

In addition to MAP, Bayesian Personalized Ranking (BPR) [11] has also been proposed as an optimization method for recommendation models. It works by optimizing an area under the curve AUC with pair-wise learning of items. Unlike MAP, however, BPR does not put a higher penalty to the mistakes at the top of the list than mistakes at the bottom of the list [4]. Since users generally consider a few top-ranked items only in the recommendation list, it is imperative to get the relevant items as high as possible in the ranking list.

For ranking tasks, there is a need to increase the relative impact of the position of the elements in the ranked list. To incorporate this information, Reciprocal Rank (RR) [12] tries to measure the position of relevant items. This method places a high focus on the first relevant item of the list. Therefore it is best suited for targeted search only, such as *"what is the best item for user"*. The RR metric does not evaluate the rest of the list of recommended items and focuses on a single item from the list. For this reason, it is not considered as a good evaluation metric for ranking tasks where users typically want a list of multiple items for comparison and selection purposes. It is denoted as *MRR* when averaged over all users in the recommender system.

$$RR_u = \frac{1}{min_i}(predvec_i \in relvec) \quad (8)$$

Another commonly used measure is Normalized Discounted Cumulative Gain (NDCG). Contrary to MAP, NDCG is a measure of relative ranking [13]; and unlike MRR, it measures the usefulness of all items based on their position in the recommended list. It follows the notion that highly relevant items are more useful when appearing earlier in the recommended list than the less relevant items. This metric is better suited for creating ranking and browsing data, but it may also pose some challenges for recommendation tasks [1] in terms of utility (usefulness/satisfaction) of relevancy and predicted relevancy scores. We discuss these issues further in Section IV.

$$NDCG_u@N = \frac{\sum_{i=1}^{N}(G(u,n,i)D(i))}{\sum_{i=1}^{N}(G'(u,n,i)D(i))} \quad (9)$$

The goal of the MAP measure is similar to the goal of the NDCG metric. Both metrics put preference on highly relevant items that have high ranks in the recommended lists. Moreover, the NDCG further evaluates the recommended list and can use the fact that *"some items are more relevant than others"*. However, it ignores the inherent problem of the recommendation that *"predictive condition can incorrectly measure the relevancy"*, i.e., it does not consider the predicted relevancy score (prediction error), and this may affect the overall recommendation quality. We suggest that predicted relevancy scores are as important as the true relevancy scores for an effective recommendation. Therefore, ranking evaluation measures should pay attention to both scores.

C. MOTIVATION FOR A NEW METRIC

As mentioned in Section II-B, there has been an increasing consensus in the recommender system community that error-based metrics are not enough for recommendation tasks, since, ultimately, users are more concerned with the relevance of the top-N items than their (predicted) rating values. Therefore, researchers have shifted towards IR-based ranking metrics rather than sticking to the originally commonly used error-based metrics. With this in mind, one may question the motivation behind proposing a new metric (rank + error-based), if recommender systems mostly care about the rank in the end. The answer to this question is that although ranking metrics are the most suitable methods to assess the top-N recommendations, ranking measures alone, i.e., without incorporating accuracy factor, might generate a less valuable ranking for top-N recommendations. This insufficient aspect of ranking measure is also due to the fact that the application of IR-based metrics for recommender systems is not straightforward [5], [14]. Firstly, the analogy of IR system is different from recommender systems mainly for the following reasons: (i) the information needs of the users — usually specified by keywords — can be approximated by similarity measure rather than prediction task; (ii) the relevance score

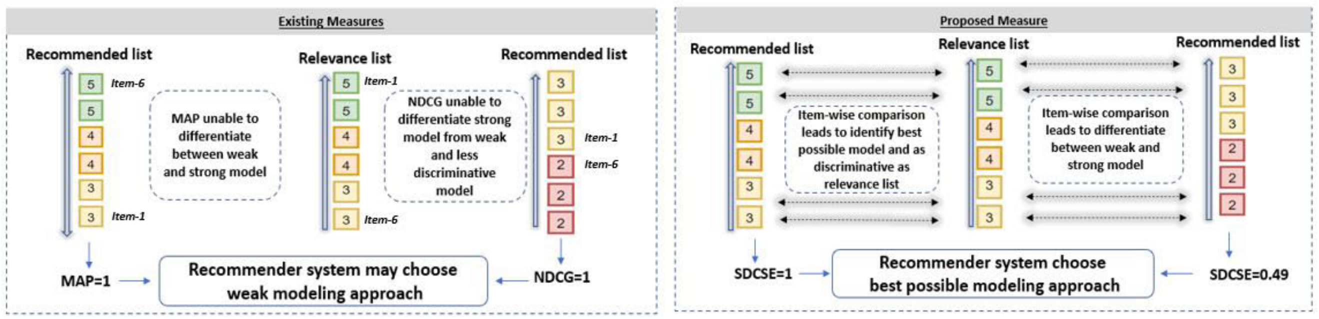

Existing Measures
Recommended list
Item-6
Item-1
MAP unable to differentiate between weak and strong model
Relevance list
Item-1
Item-6
NDCG unable to differentiate strong model from weak and less discriminative model
Recommended list
Item-1
Item-6
MAP=1
Recommender system may choose weak modeling approach
NDCG=1
Proposed Measure
Recommended list
Relevance list
Recommended list
Item-wise comparison leads to identify best possible model and as discriminative as relevance list
Item-wise comparison leads to differentiate between weak and strong model
SDCSE=1
Recommender system choose best possible modeling approach
SDCSE=0.49

FIGURE 2. Existing measures consider list-wise performance as shown by the solid arrows, single sided (for graded relevancy) and double sided (for non-graded relevancy). While proposed measure considers list wise (solid arrows) as well as an item-wise (dashed arrows) performance to keep track of accuracy of the model. Thus, enabling the recommender system to select best possible model.

is mainly independent of the users, which implies that a set of relevance score may be valid for any user; and (iii) the relevance scores are usually complete, i.e., all the relevant documents for a given query are known.

Secondly, some of the existing ranking metrics (inspired by IR-based metrics) do not incorporate the graded value of relevance scores while generating the *recommended list*. This is mainly because some measures like MAP use binary relevancy scores, making it challenging to generate rank/order within a sub-list of relevant/non-relevant items, and hence, the least relevant item is considered the same as the most relevant one. For this reason, MAP may fail to find the user's most favorite item at the top of the list.

Thirdly, some other measures, such as NDCG, exploit graded relevancy order and do not take relevance scores into account. This may lead the recommender system to apply a less powerful and less discriminative model, and hence less suitable for top-N recommendations (items may not comply with the user's true preferences), as illustrated in Figure 2. This also suggests that while NDCG might be an optimal measure for the IR domain (where all relevant documents are essential), it might be unsuitable for the recommender systems domain, in which interactions (usage/ratings) are heavily dependent upon users [5], and therefore, their true relevance scores cannot be ignored for the top-N recommendations. For example, the high-rated item may not be relevant to all users, as opposed to IR, where the top document of the recommended list is always highly relevant to the query [5]. This fact is inherent to the recommendation problem because if items were equally liked/disliked by all users, then the user-based recommendation would be an unnecessary task.

To cope with the above-mentioned issues, we propose a new evaluation measure called *standard discounted cumulative squared error (SDCSE)*. This metric tries to mitigate the challenges of both the error-based metric and rank-based metric, and put more focus on the more useful and discriminative top-N recommendations. It uses the rank and prediction error to combine the utility/ranking and accuracy in the final score. The final single value captures the differences between the user's actual ratings and predicted ratings, penalized by rank orders of the recommended list in top-N recommendations, i.e., errors at the top positions get more penalty than the at the lower positions. Penalized errors at the top increase the probabilities that more relevant and user-preferred items are placed at the top of the list. We can summarize this as follows:

- Precision-based ranking measures like MAP work well for implicit feedback, with which interactions can only be captured through success or failure (binary relevance). However, applying explicit feedback, MAP is unable to capture the fine-grained relevancy between items, which in turn may affect the order in the recommended list of items.
- Gain-based ranking measures, like NDCG, assigns a penalty to an item rating if a less relevant item is placed above a more relevant item in the list. However, NDCG does not assign any penalty if an item is less relevant to the user, or if the list is not discriminative enough (at least as discriminative as relevance list).
- SDCSE incorporates the rank as well as penalized prediction error to make it discriminative enough, such that the user's most preferred items are always at the top of the list. Thus, intuitively, SDCSE enables the recommender system to apply a fair and best possible modeling approach. In contrast, MAP and NDCG may put preferences on less accurate or weak models, as shown in Figure 2.

III. STANDARD DISCOUNTED CUMULATIVE SQUARED ERROR

The main objective of our proposed metric is to incorporate both the rank and accuracy to mitigate the challenges mentioned in Section II-B. By assigning a rank to the items, this metric ensures that a recommendation algorithm places the most relevant item at the top of the list. In addition, by including accuracy, it also considers the quality of recommendation by capturing the rating migration pattern between the relevance list and the predicted list (prediction error) and penalizing them according to their ranked position in the list.

Formally, we take relevance vector *relvec* and predicted vector *relvec* as mentioned in Section II-B. We sort the predictions *predvec* and name it as *predvec′* and also sort the *relvec* in the the order of *predvec′*, we call this relevance list as *relvec*′. For example *predvec* = [3.0, 5.0, 1.0] and *relvec* = [2.0, 3.0, 1.0] then their corresponding *predvec*′ and *relvec*′ would be equal to [5.0, 3.0, 1.0] and [3.0, 2.0, 1.0]. Afterwards, we compute the difference between *relvec*′ and *predvec*′ and square the difference, to make sure the difference is positive. We can calculate cumulative squared error *CSE* for each user as in Equation 10, where i is the index of item.

$$CSE@N = \sum_{i=1}^{N} (relvec'_i - predvec'_i)^2 \tag{10}$$

CSE gives equal weights to errors at all the ranking positions, and our objective is to minimize the error to get the most useful items at the top of the list. Thus, the proposed metric contributes by capturing the prediction error and making sure that this error is minimized at the top of the list. We calculate the discounted cumulative squared error (DCSE) to assign a higher penalty to the top error. The DCSE for a single user is given by Equation 11, where $\log_2(i+1)$ is the discount factor that progressively decreases the face value of error as the rank increases. The advantage of using logarithmic function is that it does not provide a steep decay, and hence even the last item in the list would get a significant value.

$$DCSE@N = \sum_{i=1}^{relvec'_N} \frac{(relvec'_i - predvec'_i)^2}{\log_2(i+1)} \tag{11}$$

To compare the evaluation scores among different users, they have an upper and lower bound, which can be obtained through a reference point for each score. We may achieve this reference point with the worst DCSE (WDCSE) score for each user. More specifically, WDCSE can be reached by maximizing the squared error (SE) at the top and gradually decreasing it. One of the way to obtain this is to sort the squared error used in Equation 10 in descending order, given by Equation 12, where f^{des} is a sorting function that sort the SE vector of N elements in descending order. Note that this may result in the worst reference point for each user, i.e., the maximum error at the top of the list.

$$SE^{des}@N = f^{des}(relvec'_i - predvec'_i)^2 \tag{12}$$

SE calculated in Equation 12 is used in Equation 13 to calculate the WDCSE score for each user.

$$WDCSE@N = \sum_{i=1}^{N} \frac{(SE_i^{des})}{\log_2(i+1)} \tag{13}$$

Finally, the standard DCSE (SDCSE) can be defined as the ratio of DCSE and WDCSE, as shown in Equation 14. Its final value ranges between 0 and 1. Since this is an error-based metric, the lower the value, the better the model is. In other words, $SDCSE = 0$ means no error and hence a perfect model. However, we may subtract the SDCSE score from a max SDCSE value, i.e., 1, to have an equivalent accuracy score.

$$SDCSE@N = \frac{DCSE@N}{WDCSE@N} \tag{14}$$

An overall system's error can be calculated by taking mean of SDCSE for all users as follows. Here u ∈ U and U is total number of users. Hence N_u denotes the top-N items for each user *u*.

$$MSDCSE@N = \frac{1}{U} \sum_{u=1}^{U} SDCSE@N_u \tag{15}$$

IV. EXPERIMENTAL SETTINGS AND EVALUATION METHODOLOGIES

To choose the best recommendation algorithm, more often, statistically, sound methods are desirable. In that regard, we design a number of experimental methods to analyze and compare the proposed metric with other state-of-the-art metrics. Firstly, for example, a metric with a minor migration pattern is appropriate for top-N recommendations since it correlates more with the user's preferences. Secondly, it is essential to explore how evaluation measure is sufficient to assess different algorithmic potential. Thirdly, a metric with higher discriminative power than others tends to produce more statistically significant differences among the recommender algorithms [5] but also the users, for top-N recommendations. Finally, the robustness to incompleteness is an important issue and concern in the recommendation community since the scarcity of relevance score is a common challenge for the metric reliability values [5].

Generally, for evaluation, the relevance scores are formed by the ratings in the test set, which can be treated as incomplete since they are prepared through a held-out method (test set) from the whole dataset. More specifically, the test set itself is an incomplete version of the original dataset because the users have not rated all the items and therefore need recommendations. Therefore we may say that a reliable metric should incorporate the prediction error for an accurate recommendation, have great discriminative power, and be robust to the incompleteness of the test set.

In this Section, we design a number of evaluation methods to measure the above mentioned capabilities of the metric. We discuss all these concepts in detail and explain *why and how much* they are essential for top-N recommendations. Moreover, we also discuss how SDCSE behaves and perform with respect to these concepts.

A. DISCRIMINATIVE POWER

Generally, an evaluation process is non-resistant to some degree of randomness and uncertainty. For example, the test set is a sample of the whole dataset, and its preparation (sampling formulations) often adds further random variance. Differences in the values of a metric may therefore be subject to some degree of randomness or noise. For ranking metrics, which compute the ranked list for each user, we use paired difference tests to discriminate whether the 'metric means' of two systems are statistically different or not. Ideally, we expect the set of measured values (from different evaluation measures) to have a statistically significant difference from one another to conclude how one metric is better than the other.

In our experiments, we will compare the performance of multiple recommender systems using several different datasets applying the previously described metrics. We decided to choose the method proposed by [15] and recently used by [5]. We choose Fisher's randomization test with the difference in "mean" as the statistical test [5], [16], since it provides comparatively better estimation for the $p - value$ [5], [17]. We use an approximation methodology, Monte Carlo sampling with 100,000 samples which is sufficient to compute a two-sided p-value of 0.05, as described in [5].

For assessing how discriminative a given metric is, we compute the p-value of the test among every possible pair for recommenders using that metric. While doing this paired testing, for each comparison, we consider an array of metric values (from Monte Carlo samples) for each user produced by the two systems being tested. After that, we plot the p-values sorted by decreasing order as described in [15]. This is called the p-value curve of the metric. A highly discriminative metric generates low p-values and hence, its p-value curve should be closer to the origin.

Moreover, to summarize the p-value curve in one unique value for each metric, we use the recently proposed method by [5]. They propose to compute the area under the p-curve by summing up all the p-values for the given metric. They call this value *DP* (discriminative power). Thus, the lower the value of DP, the higher the discriminative power of the metric. Note that DP depends on both the set of systems and the dataset. Therefore, we use it to compare the same systems on the same dataset.

B. ROBUSTNESS

Incompleteness has been simulated in information retrieval (IR) using unbiased random sampling techniques [8], [9]. As described in [5], incompleteness in recommender systems can be reflected as incompleteness in ratings, items, and users. However, item and user incompleteness are more related to the cold start problems, and thus it is beyond the scope of this study. As a result, in our experiment, we focus on rating incompleteness to evaluate the robustness of the proposed metric.

One of the most common biases in recommender system evaluations is the sparsity bias, which can be defined as the absence of relevance scores for user-item pairs involved in a recommendation [14]. We decided to use the method proposed by [5] to assess robustness. They assessed the robustness of different ranking metrics to the rating's incompleteness by using random test set samples. We define different test sizes starting from 10 percent to 100 percent of the size of the original test set, and we take 50 random samples of each size from the test set as described in [5]. Given a set of recommenders and a particular metric we first rank these systems for each test set sample. Second, we compute Kendall's rank correlation coefficient of each system ranking with respect to the original test set's ranking. Third and finally, by averaging the rank correlation of the samples having the same size, we obtain a final estimate of the robustness of a metric for each test size. For system comparison, the smaller the test size (i.e., the simulated sparsity is more aggressive), the lower the expected correlation. We suggest that a metric is more robust than another if it has a higher average correlation with itself (original test set) as the test set is reduced.

C. ASSESSING ITEMS MIGRATION PATTERN

As mentioned in Section II-B, existing measures either use precision-based scores or compute cumulative gain for generating rank over items. However, we emphasize that while calculating precision or generating ranking over items, existing measures may incorporate some error factors into the final score. For example, the ranking/precision function (MAP, NDCG) that usually takes *relvec* and *predvec* vectors may generate a perfect score leading to a perfect mapping between a relevance list and a recommended list. However, we argue that mapping from the relevance list to the recommended list may have three types of item mappings, up-sell, down-sell and perfect-sell (us, ds, ps) resulting in three error functions. These functions are mentioned in Equation 16, Equation 17 and Equation 18.

$$f_{us}(k) = \begin{cases} 1 & \text{if } (k)predvec > (k)relvec + \lambda \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

$$f_{ds}(k) = \begin{cases} 1 & \text{if } (k)predvec < (k)relvec + \lambda \\ 0 & \text{otherwise} \end{cases} \tag{17}$$

$$f_{ps}(k) = \begin{cases} 1 & \text{if } (k)predvec = (k)relvec \underline{+} \lambda \\ 0 & \text{otherwise} \end{cases} \tag{18}$$

Error functions take a list of N items for each user and tag binary values to items depending upon the conditions. We call these error functions as f_{us} and f_{ds}. For a given predictive vector and relevance vector, denoted as $(k)predvec$ and $(k)reldvec$ relatively, f_{us} assigns '1' to an item at position k if its predicted score is relatively higher than the relevance score, including a margin factor λ. Else, f_{ds} defined as '1' if it is predictive score is relatively lower than relevance score, plus a margin factor λ (de-graded relevancy). See Figure 1 for an illustration. Here λ controls the degree of variance between relevance score and predictive score, estimated from score distribution. It also serves as a tool to incorporate the inherent biasing factor resulting from explicit ratings. Finally, f_{ps} is the function that assigns 1 to an item if it does not change its position plus a margin factor $\underline{+}$ in the recommended list. Lower values of f_{us} and f_{ds} quantify that recommended list correlates better with the user preferences as per the relevance list.

Finally, Equation 19 and Equation 20 compute the final up-sell and down-sell score ($us@N$ and $ds@N$) for each user at top-N items. Here, N_u denotes the top-N items for each user u, and the scores for $us@N$ and $ds@N$ are within [0,1] interval.

$$us@N = \frac{1}{U}\sum_{u=1}^{U} f_{us}@N_u \quad (19)$$

$$ds@N = \frac{1}{U}\sum_{u=1}^{U} f_{ds}@N_u \quad (20)$$

In a real-world scenario, the above-mentioned item mappings and error factors are inevitable and cannot be avoided during recommendation modeling approaches and optimization. However, we argue that these error factors should be reflected in the final evaluation score to better differentiate and approximate the strengths and weaknesses of the modeling approaches. The challenge is how to capture and incorporate these errors factors efficiently while evaluating the models or systems. Existing measures either ignore this error information (with MAP) or are inefficient at measuring it (with NDCG). Specifically, *us* and *ds* can play a crucial role in ordering items in the list and hence attention should be paid to capture this difference in ratings for efficient and useful top-N recommendations. For example, *us* may bring relevant but less preferred items (as shown in Figure 2 and Figure 1) at the top of the list and hence should be penalized depending upon the position. The higher the error in the position, the more should be the penalty. Similarly, as a result of *ds*, relevant and most preferred items are either missed from the recommended list or are present at the bottom of the list and thus, the user may not find his favorite items at the top of the list resulting in a user's low interest and low engagement.

Furthermore, if all or most of the items are predicted with relatively low ratings while maintaining the same order as in the relevance list, still it is not desirable since items may move below the relevance threshold (if fixed) and drop out from the top-N list. See Figure 1.

D. COMPARATIVE STUDY

This section suggests comparing the proposed metric with existing metrics using state-of-the-art recommendation algorithms, including point-wise and pair-wise methods and contextual and non-contextual approaches. The objective of this comparison is two folds: 1) to compare the accuracy/precision score and the statistical pattern between proposed and existing measures, and 2) to compare the algorithmic potential towards generating effective relevancy scores and its relationship with the evaluation measures.

Here, we list the latest state-of-the-art methods we use in our experiment.

- ***Matrix factorization (MF):*** Matrix factorization methods [18] learn low rank-decomposition of user-item interaction matrix by minimizing the square loss function. We specifically used this technique as a baseline to compute the preference degree by the product of user features and item features.
- ***Latent Cross Neural Network (LC-NN):*** Latent Cross methods [19] take contextual data as a part of the input and treat it differently. We use this method with CNN and point-wise loss and take the dot product of hidden vector and context vector and hence the name is Latent Cross.
- ***Bayesian personalized ranking (BPR):*** Bayesian personalized ranking (BPR) [11] is a *loss* function and optimization method (a variant of stochastic gradient descent), which accepts inputs in the form of pairs/triplets, i.e., positive and negative. Thus, it takes the dot product of user features with all available (positive and negative) features, making it the right choice for personalized ranking.
- ***Neural collaborative filtering (NCF):*** We use a neural network to learn features, i.e., user-item interactions along with contextual vectors and then use MLP with learned features for final prediction as described in [20].

It is essential to mention that our objective is not to prove the best modeling approach through extensive hyperparameter tuning and pre-training tweaks. Nevertheless, we demonstrate how existing evaluation measures perform, assess and approximate the strengths of different modeling approaches and identify the added value of our proposed metric in extracting the model's potential in status-quo. Status-quo is the same for all involved evaluation measures to avoid any biases. All the comparisons have been made at L levels of relevancy thresholds, i.e., $L = \{\text{Low-High, Medium-High, High}\}$ to capture the fine-grained performance of the metric for the varying levels of relevancy of items.

E. DATASETS AND TRAINING DETAILS

In our experiments, we use a variety of datasets from different domains, including Movies (MovieLens), Books (Book-Crossings), Research Papers feedback, products (Amazon accessories) and Rental services (clothing brand-RentTheRunway). Here, the *MovieLens 25M (ml-25m)* dataset is the latest and more stable version (released in 12/2019), with ratings 1 to 5. The books dataset [21] includes ratings (1-10) for books. The research paper dataset (RP) [22] has the user's ratings (1-10) for research papers. The amazon dataset (AMZ) [23] includes ratings (1-5) for different accessories like shoes and jewelry. Finally, the *RentTheRunway* (RR) [24] is one of the largest rental platforms for women's clothing, where ratings range from 1 to 10.

We also created implicit versions of the above-mentioned data sets. All items rated by the user u in the test set with a value below a certain relevance threshold τ is considered 'Not-liked' for that user and above τ are 'liked.' We derived different thresholds for different data sets depending upon rating distribution. For example, it is common to set τ to 4 in ratings ranging from 1 to 5. To compare the modeling algorithms, we use two types of model learning: point-wise [25], [26] and pair-wise [27], [28]. Moreover, for each type, we use context-aware methods when the context is available with the dataset. Otherwise, we use non-context state-of-the-art methods. Here, the context information includes the user's profile information, such as age, height and occupation.

The dataset preparation method is different for these two learning methods since a point-wise method accepts individual inputs, while a pair-wise learning method accepts inputs in the form of pairs or triplets. For point-wise learning, we split the dataset into 70 % training set and 30% test set. For pair-wise learning, we first create the pairs/triplets, in which each user is coupled with positive items (extreme positive with ratings $\geq$ 5) and randomly chosen negative items (extreme negative with ratings $\leq$ 2). This method of creating a triplet is specifically helpful for learning discriminating features.

The training involves two models, embedding learner and regressor. For the embedding learner, we use a convolutional network consisting of three convolutional and 1×1 average-pooling layers. The network configuration (ordered from input to output) consists of filter sizes $\{1, 2, 2\}$ and feature map dimensions $\{150, 500, 500, 256\}$, where the 150 vector is the embedding dimension (input size) and the 256 vector is the embedded representation of the network followed by two fully-connected hidden layers, each with 500 nodes and a Relu activation function. The final output layer is with 256 nodes and this 256 vector is the final representation of embeddings coming from each instance of the triplet network.

We apply an L_2 normalization to the embeddings before feeding them into the triplet loss function. Applying normalization like this can be compared to the advantage of cosine similarity to Euclidean distance. The squared Euclidean distance between normalized vectors is proportionate to their cosine similarity, so the squared Euclidean distance is guaranteed to be within the range [0, 4] (cosine similarity value). The training is done in batches with a batch size of 64 and runs over ten epochs. The embedding layer dropout is fixed at 0.05, whereas the layer dropout is in order $\{0.5, 0.5, 0.25\}$ from hidden to the output layer. The learning rate is 0.005 and the momentum is 0.9. The model is trained using back-propagation with ADAM [29]. During each training pass, the embeddings are evolved and improved by using triplet hinge distribution loss at the end of the network as mentioned in loss [30]. Finally, for final prediction, the embedded learner's learned embeddings are passed to a regressor, 1-layer feed-forward neural network.

V. RESULTS AND COMPARATIVE STUDY

In this section, we show the results for all the methods explained in Section IV. Furthermore, we discuss how SDCSE performs compared to the other evaluation measures for all the involved methods.

A. DISCRIMINATIVE POWER

In this section, we show the results of discriminative power (DP) for all the metrics. We present the values of DP (which amounts to the area under the p-value curve) in Table 1. Overall, SDCSE has higher discriminative power than the both MAP and NDCG on all datasets except *ML*.

TABLE 1. The DP values (lower is better) among all datasets. Overall, SDCSE shows lower DP values.

Evaluation Measures	ML	RR	BOOK	RP	AMZ
MAP	0.8883	2.9790	0.3100	0.6007	0.0703
NDCG	**0.2415**	0.9509	0.0001	0.77	0.2250
SDCSE	1.7311	**0.0003**	**0.0001**	**0.0002**	**0.0041**

NDCG, on the other hand, performs better than MAP in three datasets. This is consistent with the findings in one of the recent work [5], in which the authors concluded that NDCG has most discriminative power than any other metric, including MAP.

The metrics that use graded relevance (i.e., NDCG) seem to discriminate better among systems [5]. For instance, when two systems a and b find one relevant item each on position 5 for the user, only the graded metrics will produce different performance values when the item found by system a was rated higher than the one found by b. However, SDCSE takes this concept further by enabling it to discriminate top-N recommendations for each user. For example, when three relevant items at positions 3, 4 and 5 (having different relevance scores) get the same rank (due to the same prediction), SDCSE will produce different values for each by considering their respective prediction errors. Moreover, it also provides a more diverse or dynamic range of metric values than a possible set of relevant graded values. Overall, we can conclude that SDCSE has higher discriminative power than existing commonly used metrics.

B. ROBUSTNESS

In this section, we present the results of robustness for all metrics. In Figures 3 we can see that all the metrics are reasonably robust to random rating sparsity since the correlation is higher than 0.75, even after removing half of the test set. Although the results are not very decisive, we can capture some trends. SDCSE and MAP perform better than NDCG on two datasets. Whereas NDCG performs better on three datasets, but the difference is not very significant. In this regard, NDCG can be viewed as a more robust metric than the other two, which is also in line with most recent findings [5]. Interestingly, NDCG performs best in RR, the most unstable dataset, whereas SDCSE and MAP have similar robustness. Another interesting finding from a previous study is that NDCG shows less robustness to popularity bias [5]. We also observed this behavior in $ML(25m)$ and AMZ datasets, where higher ratings form the most concentration of the overall ratings. Applied on these datasets, SDCSE and MAP are more robust than NDCG. Overall, we can conclude that SDCSE has a competitive robustness property compared to existing commonly used metrics.

C. ASSESSING ITEMS MIGRATION PATTERN

In this section, we present the results of different migration patterns and show how well different evaluation measures assess and reflect this information in the final score.

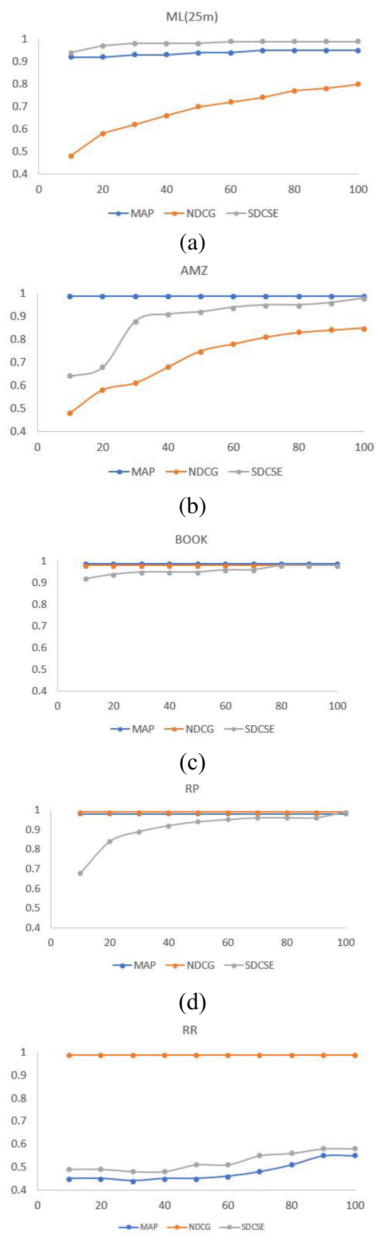


FIGURE 3. Comparison of robustness between proposed metric and state of the art methods for data sets AMZ, BOOK, RP and RR is shown in (a), (b), (c), (d), and (e) respectively.

Following are some of the scenarios comparing SDCSE scores with existing measures scores. We have taken these scenarios from the data set and simulated them to compare the proposed metric effectiveness. We use p to denote predictive score and y to denote their corresponding ground truth values, symbols $\langle\rangle$ denote the subset of predictive and relevance vectors for demonstration purposes.

- In this scenario, MAP is maximum, although some least relevant items are at the top and most relevant is at the bottom of the list, but MAP is unable to capture this difference due to binary relevancy, see Section II-B. However, the proposed metric, SDCSE, penalizes the bad prediction (prediction error) and does not assign a full score.

$$MAP = \begin{cases} y = \langle \underline{3.0, 3.0}, 4.0, \underline{3.0}, 5.0\rangle \\ p = \langle 5.0, 5.0, 4.0, 4.0, 4.0\rangle \\ \text{MAP} = \{1\} \\ \text{SDCSE} = \{0.15\} \end{cases}$$

- In the following use case, we have compared our proposed metric with NDCG and found that NDCG may bring some bad items in the list and still assign the maximum score. This is due to the fact that as long as the order is the same in both the lists, predicted and relevant, NDCG assigns a full score. Moreover, if the top item is most relevant and the rest of the items are less relevant, NDCG assigns a full score (because more relevant precedes the less relevant), as shown in the following scenario. Moreover, based on the recommended list, since all items are relevant, the user may get a complete list of recommended items that are not of his interest. On the other hand, SDCSE captures this behavior and assigns a penalized score.

$$NDCG = \begin{cases} y = \langle 4.0, \underline{3.0, 2.0, 2.0, 1.0}\rangle \\ p = \langle 5.0, 5.0, 5.0, 4.0, 4.0\rangle \\ \text{NDCG} = \{1\} \\ \text{SDCSE} = \{0.29\} \end{cases}$$

- As opposed to the above scenario, the relevance list may have all highly relevant items and therefore, NDCG assigns full score, but there is a contradiction between relevance list and recommended list and NDCG does not capture this. Therefore, the user may get the only top item for recommendation and may lose interest. Still, SDCSE assigns a penalized score to this use case.

$$NDCG = \begin{cases} y = \langle 5.0, 5.0, 4.0, 4.0, 4.0\rangle \\ p = \langle 5.0, \underline{3.0, 2.0, 1.0, 1.0}\rangle \\ \text{NDCG} = \{1\} \\ \text{SDCSE} = \{0.35\} \end{cases}$$

For implicit feedback, NDCG also does not penalize the non-relevant items as long as the descending order is maintained, as shown in the following example.

$$NDCG = \begin{cases} y = \langle 1.0, 1.0, 1.0, 1.0, 1.0\rangle \\ p = \langle 1.0, 1.0, 1.0, \underline{0.0, 0.0}\rangle \\ \text{NDCG} = \{1\} \\ \text{SDCSE} = \{0.35\} \end{cases}$$

NDCG uses the predicted order to rank the items in the relevance list, which may be an appropriate choice for document retrieval and query comparison systems where accuracy is maximum as long as items/queries are arranged in descending order of relevance. However, in the case of recommender systems this metric may pose some challenges as it assigns the same score to the users, although one user might have more relevant list of items than the other user. This means that in the above two cases, $\langle 4.0, \underline{3.0, 2.0, 2.0, 1.0}\rangle$ and $\langle 5.0, \underline{5.0, 5.0, 4.0, 4.0}\rangle$, are treated as equivalent with NDCG.

In recommendation systems, the problem mentioned earlier may further trigger two problems. First, if there is a conflict between relevance ratings and predicted ratings, and the evaluation measure does not capture it (though both follow descending order), then it might enable the recommendation system to consume the model with less predictive accuracy. This, in turn, might generate similar features for potentially two different users or cannot differentiate well between a high relevant score and a less relevant score. Second, while predicting items for which users have not rated them yet, which is mostly the case with recommendation systems, less relevant items might be presented to the user as the most relevant ones. Similarly, recommendation systems might ignore some of the most relevant recommendations if the ratings are below the relevance threshold.

More precisely, an evaluation measure should estimate any rating pattern migration, i.e., up-sell or down-sell of items, which can in turn be considered as an approximation of modeling strengths/weaknesses, and the final score should reflect this behavior. Our study shows that existing evaluation measures are unable to capture this behavior, which in the long run, might result in decreasing the quality of recommendation systems. Fortunately, by integrating the error metrics with the ranked approach, such a capability of the model can be measured efficiently with the proposed SDCSE.

Several qualitative aspects of the proposed evaluation metric are associated with the accuracy score that is worth discussing.

First, this metric is appropriate to keep the *interestingness* of the items by preserving the user's preference structure. This structural property is enforced by extracting and penalizing the error. By assigning a higher penalty to the top error, it ensures that the user's desired item has a higher priority along with the relevance order. MAP cannot capture this scenario while NDCG also ignores rating patterns and use ordering instead, as explained in the above use-cases.

Second, SDCSE is beneficial for new users who are exposed to only a limited number of items and are in the process of exploring items. For quick and effective engagement, it is more important to pay attention to down-sell and up-sell. For example, if the user's most favorite items are at the bottom of the recommended list due to a down-sell (MAP) or the metric being unable to penalize non-relevant items (NDCG), it may cause a bad experience for the user at the start of his/her system interaction. Similarly, if the user is presented with items that he/she does not like at all, it may also ruin the user experience.

Third, a recommender system may fail to present the user's potential favorite items to him/her, if they are below the recommended list's relevance threshold. For example, if the user is a high-value customer, e.g., in the e-commerce industry, this might result in a challenge, such as losing the customer due to constant frustration and revenue loss. Existing ranking measures, such as NDCG, are unable to help to avoid this situation. In contrast, SDCSE enforces both relevancy and user preferences in a single measure, thus capturing this potential problem.

In summary, we may say that up-sell and down-sell measurements of graded ratings (prediction error) are imperative to improve the general quality of recommendations, in addition to ranked order for top-N recommendations. Moreover, by integrating both ranking and error-based evaluation approaches, our proposed measure can assess the utility and the accuracy of recommendations simultaneously.

TABLE 2. Ratings migration (us-ds) for each model among different data sets. Lower the value, better the model.

Models	ML	RR	BOOK	RP	AMZ
MF	**0.87**	**0.70**	**0.85**	**0.65**	**0.65**
BPR	0.91	0.74	0.87	0.77	0.78
NCF	0.95	0.73	-	-	-
LC-NN	0.90	**0.69**	-	-	-

D. COMPARATIVE STUDY

In this section, we compare how evaluation measures are good at assessing the algorithm's potential. We perform our experiment for different levels of relevance threshold where $Low-High \in 1, 2, 3, 4, 5$, $Med-High \in 3, 4, 5$ and $High \in 4, 5$. We summarize the comparative analysis as follows:

- We observed that the proposed metric, SDCSE, assigns a higher score to MF at low-high and mid-high threshold and LC-NN at the high threshold, as shown in Figure 4. This is because MF and LC-NN have the least up-sell and down-sell error/migration as discussed in Section IV-C. This phenomenon supports that SDCSE helps a recommender system achieve the best match of users' preferences. We also support this argument statistically by extracting the up-sell and down-sell score distribution from the datasets that we use in our analysis, as shown in Table 2. This behavior is also well-aligned with the objective of our proposed metric, i.e., top items of the list must match the user's preferred items, having the least us/ds score (prediction error). In contrast, NDCG ranks BPR relatively higher in the low-high and med-high threshold, proving it the optimum approach. However, as shown in Table 2, statistically, NDCG has the higher us/ds error than MF. Similarly, MAP assigns the lower score (except with the Books dataset) to MF, though MF has a lower us-ds error in most datasets. This shows that neither NDCG nor MAP can assess the weaknesses of a predictive model.
- In *RentTheRunway* dataset, NDCG assigns the highest score to NCF. However, as depicted in Table 2, statistically, NCF has the highest us/ds error than all other models. Again, this modeling weakness is not captured by NDCG since it does not take the rating scores into account and uses the order instead. As opposed to NDCG and MAP, SDCSE's ordering of models aligns

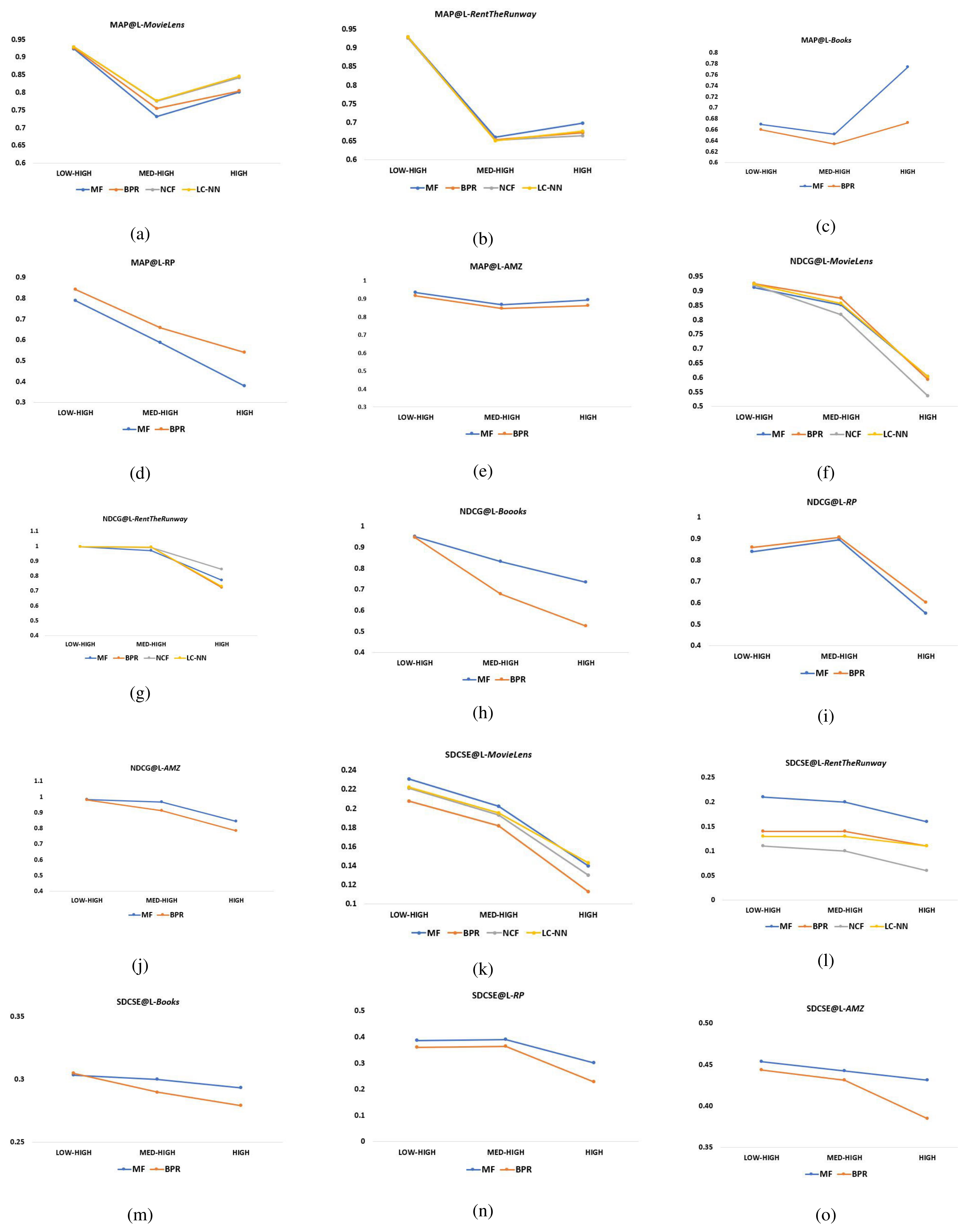


FIGURE 4. **Comparison of NDCG (a:e), MAP (f:j) and SDCSE (k:o) among different models at different level of relevancy threshold and for different datasets.**

well with the distribution ratings, i.e., the model with the least prediction error set to be the best, while the model with the most prediction error is at the lowest position, as shown in Figure 4.

- NDCG and SDCSE both show the declining curve of performance from low to high relevancy, which shows the statistical consistency of pattern for proposed metric with NDCG, since both deals with cumulative function, though, NDCG deals with gain metric (discrete values) while SDCSE deals with error metric (continuous values). However, MAP deals with conditional function (bool values) and it follows a different pattern from both the metrics.

- It is important to mention again that our objective is to demonstrate how evaluation measures perform and contribute towards extracting and quantifying the algorithm's potential. Therefore, we noticed that SDCSE assigns a relatively lower model score than NDCG and MAP scores while exploring this relationship. This is because the former put strict penalty to the prediction error, while NDCG is a relevancy-gain based metric and does not penalize the prediction error and hence may assign a full score to the list while evaluating it, as explained in Section II-B and Section II-C. Similarly, MAP cannot differentiate between low and high relevancy and may assign a maximum score to both the lists, one with the highest relevant items and the other with low relevant items. This is why existing measures may not differentiate between good and weak models and assign the same score to a less relevant recommended list and a more relevant recommended list. On the contrary, SDCSE may help select a more preferred list over a less preferred list since it considers the modeling capability through prediction error.
- NDCG and MAP operate at a list level and ignore the individual score values. Therefore, these measures may not perform well in a real-world scenario where prediction error while achieving user's preference structure is inevitable and visibility of prediction score to end-user is encouraged. However, SDCSE operates at the item level (prediction error) and the list level (rank) and will never assign a full score unless it is a perfect world. Therefore it is the most appropriate measure to evaluate the modeling performance in the real world scenario.
- Another property of SDCSE is its ability to differentiate models with a considerable margin. NDCG and MAP may generate an overlapping score since they consume a *fixed* set of discrete relevance scores, for example, 1 to 5. However, SDCSE may deal with continuous scores due to the prediction error (at item level). For example, using the prediction error for each item (each user may have different error distribution in the list) in cumulative function at the list level may result in a variable and non-overlapping score for each user. Therefore, each user may contribute to a non-overlapping score towards achieving a unique and varying distribution for each model.
- SDCSE also works well with the implicit feedback as shown in Table 3, where we have used this metric with binary versions of the datasets. However, this experiment is designed to compare and to show the feasibility and generality of the proposed metric instead of showcasing the best modeling approach.

TABLE 3. SDCSE score for implicit feedback among different datasets.

Models	ML	RR	BOOK	RP	AMZ
MF	**0.47**	**0.17**	**0.29**	0.38	**0.47**
BPR	0.45	0.14	0.27	**0.45**	0.45
NCF	0.46	0.13	-	-	-
LC-NN	0.44	0.11	-	-	-

E. LIMITATIONS OF PROPOSED METRIC

Generally, all metrics have some limitations, and so does the metric proposed in this paper. First, we use accuracy in combination with rank and for this reason, like other metrics, SDCSE uses true relevance score in the testing and evaluation phase. This could be a limitation since we cannot guarantee that true relevance scores are always available. A possible way to cope with this is to use inferred average precision (InfAp) for approximation when the true relevance values are missing or are incomplete.

Second, although the proposed metric is sufficiently robust and has better discriminative power than all the metrics (more suitable for top-N recommendations), it is not as robust as the more commonly used state-of-the-art metric, NDCG. Nevertheless, SDCSE is more robust than NDCG against popularity biases, which shows its potentials. In our future study, we will use this finding further to investigate better methods to improve the robustness of the proposed metric.

VI. CONCLUSION AND FUTURE WORK

In this paper, we proposed a new error-based ranking evaluation approach to improve the way recommendation algorithms are evaluated. We discussed and showed that existing measures might incorporate several error factors when computing the accuracy of the evaluated models, which might, as a result, hurt the overall quality of recommendations. The proposed approach mitigates the challenges of existing measures by evaluating both the accuracy and utility of the recommendation in the same metric. Furthermore, it deals with these challenges by exploiting the predictive accuracy for each item with respect to its ranking position in the list. Our experiments demonstrated that the proposed metric improves the modeling strength assessment and enables the recommender system to utilize the best possible model. To evaluate our approach, we conducted a detailed empirical analysis at a user and model level to showcase the comparative analytical aspects and the added value of the proposed metric. Our results show that the proposed metric outperforms the existing metrics at the user level by assigning more realistic and penalized scores to the user's items. Moreover, by capturing the modeling strengths at the item and list levels, SDCSE can discriminate the models from each other better than the state-of-the-art metrics. All this demonstrates the applicability, generality, and usefulness of the new metric. For future work, we plan to extend our work by incorporating other measures, which are not directly related to accuracy, such as diversity and coverage.

REFERENCES

[1] T. Silveira, M. Zhang, X. Lin, Y. Liu, and S. Ma, "How good your recommender system is? A survey on evaluations in recommendation," *Int. J. Mach. Learn. Cybern.*, vol. 10, no. 5, pp. 813–831, May 2019.

[2] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, "Evaluating collaborative filtering recommender systems," *ACM Trans. Inf. Syst.*, vol. 22, no. 1, pp. 5–53, 2004.

[3] A. Gunawardana and G. Shani, "Evaluating recommender systems," in *Recommender Systems Handbook*. Boston, MA, USA: Springer, 2015, pp. 265–308.

[4] P. Cremonesi, Y. Koren, and R. Turrin, "Performance of recommender algorithms on top-N recommendation tasks," in *Proc. 4th ACM Conf. Rec. Syst. (RecSys)*, 2010, pp. 39–46.

[5] D. Valcarce, A. Bellogín, J. Parapar, and P. Castells, "Assessing ranking metrics in top-N recommendation," *Inf. Retr. J.*, vol. 23, no. 4, pp. 411–448, Aug. 2020.

[6] F. Ricci, L. Rokach, and B. Shapira, "Introduction to recommender systems handbook," in *Recommender Systems Handbook*. Boston, MA, USA: Springer, 2011, pp. 1–35.

[7] X. He, T. Chen, M.-Y. Kan, and X. Chen, "TriRank: Review-aware explainable recommendation by modeling aspects," in *Proc. 24th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2015, pp. 1661–1670.

[8] C. Buckley and E. M. Voorhees, "Retrieval evaluation with incomplete information," in *Proc. 27th Annu. Int. Conf. Res. Develop. Inf. Retr. (SIGIR)*, 2004, pp. 25–32.

[9] E. Yilmaz and J. A. Aslam, "Estimating average precision when judgments are incomplete," *Knowl. Inf. Syst.*, vol. 16, no. 2, pp. 173–211, Aug. 2008.

[10] Y. Yue, T. Finley, F. Radlinski, and T. Joachims, "A support vector method for optimizing average precision," in *Proc. 30th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr. (SIGIR)*, 2007, pp. 271–278.

[11] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian personalized ranking from implicit feedback," in *Proc. 25th Conf. Uncertainty Artif. Intell.*, 2009, pp. 452–461.

[12] E. M. Voorhees, "The TREC-8 question answering track report," in *Proc. TREC*, vol. 99, 1999, pp. 77–82.

[13] K. Järvelin and J. Kekäläinen, "IR evaluation methods for retrieving highly relevant documents," in *Proc. ACM SIGIR Forum*, New York, NY, USA, 2017, vol. 51, no. 2, pp. 243–250.

[14] A. Bellogín, P. Castells, and I. Cantador, "Statistical biases in information retrieval metrics for recommender systems," *Inf. Retr. J.*, vol. 20, no. 6, pp. 606–634, Dec. 2017.

[15] T. Sakai, "Evaluating evaluation metrics based on the bootstrap," in *Proc. 29th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr. (SIGIR)*, 2006, pp. 525–532.

[16] R. J. Tibshirani and B. Efron, *An Introduction to the Bootstrap* (Monographs on Statistics and Applied Probability), vol. 57. New York, NY, USA: Chapman & Hall, 1993, pp. 1–436.

[17] J. Parapar, D. E. Losada, M. A. Presedo-Quindimil, and A. Barreiro, "Using score distributions to compare statistical significance tests for information retrieval evaluation," *J. Assoc. Inf. Sci. Technol.*, vol. 71, no. 1, pp. 98–113, Jan. 2020.

[18] F. Vasile, E. Smirnova, and A. Conneau, "Meta-prod2vec: Product embeddings using side-information for recommendation," in *Proc. ACM RecSys*, 2016, pp. 225–232.

[19] A. Beutel, P. Covington, S. Jain, C. Xu, J. Li, V. Gatto, and E. H. Chi, "Latent cross: Making use of context in recurrent recommender systems," in *Proc. ACM WSDM*, 2018, pp. 46–54.

[20] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proc. 26th Int. Conf. World Wide Web*, 2017, pp. 173–182.

[21] C.-N. Ziegler, S. M. McNee, J. A. Konstan, and G. Lausen, "Improving recommendation lists through topic diversification," in *Proc. 14th Int. Conf. World Wide Web (WWW)*, 2005, pp. 22–32.

[22] J. Beel, S. Langer, B. Gipp, and A. Nürnberger, "The architecture and datasets of docear's research paper recommender system," *D-Lib Mag.*, vol. 20, nos. 11–12, pp. 1–11, 2014.

[23] J. Ni, J. Li, and J. McAuley, "Justifying recommendations using distantly-labeled reviews and fine-grained aspects," in *Proc. Conf. Empirical Methods Natural Lang. Process., 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP)*, 2019, pp. 188–197.

[24] R. Misra, M. Wan, and J. McAuley, "Decomposing fit semantics for product size recommendation in metric spaces," in *Proc. 12th ACM Conf. Rec. Syst.*, Sep. 2018, pp. 422–426.

[25] X. He, H. Zhang, M.-Y. Kan, and T.-S. Chua, "Fast matrix factorization for online recommendation with implicit feedback," in *Proc. 39th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2016, pp. 549–558.

[26] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative filtering for implicit feedback datasets," in *Proc. 8th IEEE Int. Conf. Data Mining*, Dec. 2008, pp. 263–272.

[27] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian personalized ranking from implicit feedback," 2012, *arXiv:1205.2618*.

[28] R. Socher, D. Chen, C. D. Manning, and A. Ng, "Reasoning with neural tensor networks for knowledge base completion," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 926–934.

[29] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[30] C.-K. Hsieh, L. Yang, Y. Cui, T.-Y. Lin, S. Belongie, and D. Estrin, "Collaborative metric learning," in *Proc. 26th Int. Conf. World Wide Web*, Apr. 2017, pp. 193–201.

SOFIA AFTAB received the M.S. degree in IT (data mining) from the National University of Science and Technology, Pakistan. She is currently pursuing the Ph.D. degree in machine learning with the Norwegian University of Science and Technology, Trondheim, Norway. From 2012 to 2018, she worked as a Data Scientist for different industrial sectors in the domain of recommender systems. Her research interests include information retrieval, deep learning, and text mining.

HERI RAMAMPIARO is currently the Head of Department and a Professor with the Department of Computer Science, Norwegian University of Science and Technology (NTNU), Trondheim, Norway. Previously, he was the Head of the Data and Artificial Intelligence (DART) Research Group. He has been central in the establishment of the Telenor–NTNU AI-Lab, AI Research Center, NTNU (now Norwegian Open AI-Lab), for which he was a NTNU's Scientific Coordinator. His current research interests include machine learning, information retrieval, and data/text mining.