Journal of the Brazilian Computer Society, 2025, 31:1, doi: 10.5753/jbcs.2025.5426

Beyond Recommendations: Intrinsic Evaluation Strategies for Item Embeddings in Recommender Systems

Pedro R. Pires [**Federal University of São Carlos** | ***pedro.pires@dcomp.sor.ufscar.br***]
Tiago A. Almeida [**Federal University of São Carlos** | ***talmeida@ufscar.br***]

Intelligent Systems and Data Science Laboratory (LaSID), Department of Computer Science (DComp), Federal University of São Carlos (UFSCar), Rod. João Leme dos Santos, Km 110, Sorocaba, SP, 18052-780, Brasil.

Received: 15 January 2025 • **Accepted:** 19 May 2025 • **Published:** 18 August 2025

Abstract With the constant growth in available information and the widespread adoption of technology, recommender systems have to deal with an ever-growing number of users and items. To alleviate problems of scalability and sparsity that arise with this growth, many recommender systems aim to generate low-dimensional dense representations of items. Among different strategies with this shared goal, e.g., matrix factorization and graph-based techniques, neural embeddings have gained significant attention in recent literature. This type of representation leverages neural networks to learn dense vectors that encapsulate intrinsic meaning. However, most studies proposing embeddings for recommender systems, regardless of the underlying strategy, tend to ignore this property and focus primarily on extrinsic evaluations. This study aims to bridge this gap by presenting a guideline for assessing the intrinsic quality of matrix factorization and neural-based embedding models for collaborative filtering. To enrich the evaluation pipeline, we adapt an intrinsic evaluation task commonly used in Natural Language Processing and propose a novel strategy for evaluating the learned representation in comparison to a content-based scenario. We apply these techniques to established and state-of-the-art recommender models, discussing and comparing the results with those of traditional extrinsic evaluations. Results show how vector representations that do not yield good recommendations can still be useful in other tasks that demand intrinsic knowledge. Conversely, models excelling at generating recommendations may not perform as well in intrinsic tasks. These results underscore the importance of considering intrinsic evaluation, a perspective often overlooked in the literature, and highlight its potential to uncover valuable insights about embedding models.

Keywords: Embeddings, Intrinsic Evaluation, Qualitative Evaluation, Recommender Systems, Similarity Tables, Intruder Detection, Autotagging

1 Introduction

Due to technological development and growing popularity, people are regularly exposed to an increasing volume of information. Consequently, a single user can consume thousands of films, songs, books, and other items. However, of this massive amount of information, it is expected that only a portion is of their interest. In this context, recommender systems play an important role and are essential to the business of many companies. From e-commerce stores to streaming services, it is rare to find a large technology company or online service that does not have a system for this purpose [Bobadilla *et al.*, 2013].

Among different types of recommender systems, collaborative filtering (CF) remains one of the most widely used approaches. CF algorithms work by learning users' preferences based on other similar users and yielding recommendations according to previously consumed items. Pioneer CF recommender systems represented items as sparse vectors of consumption and used this representation to calculate similarities, recommending in a neighborhood-based manner. However, with the rapid expansion in the number of users and items, this form of representation began to encounter significant limitations [Khsuro *et al.*, 2016], such as:

- *Sparsity*: modern recommender systems have to deal with interaction patterns that follow a power-law distribution, often resulting in interaction matrices that extremely sparse;
- *Scalability*: as the number of users and items grows, the vectors representing these entities can become quite large. This increases the demand for greater storage and processing capacity.

To address these challenges, researchers have focused on representing users and items in a much smaller dimensional space [Sarwar *et al.*, 2000]. In this context, two prominent techniques have emerged: matrix factorization [Koren *et al.*, 2009] and neural networks [Zhang *et al.*, 2019]. In the latter, neural embedding models inspired by Natural Language Processing (NLP) advancements have gained significant attention [Ozsoy, 2016]. A key advantage of embeddings in NLP lies in their ability to encapsulate intrinsic meaning, i.e., knowledge embedded within the representation that extends beyond the training data. In the context of recommender systems, this translates to the potential of leveraging item embeddings for various tasks beyond mere recommendation.

Despite the promising nature of embeddings, much of the existing research has concentrated on extrinsic evaluations, often overlooking the intrinsic properties of the learned representations. It is well known that the performance of embeddings in downstream tasks does not always align with their intrinsic quality [Schnabel *et al.*, 2015]. Additionally, embeddings can behave differently across various tasks based on their training [Qiu *et al.*, 2018].

Although the primary objective of a recommender system is to provide high-quality recommendations, many other tasks can be supported by the use of high-quality embeddings. Automatic feature prediction, knowledge discovery, and user or item clustering [Lu *et al.*, 2015; Hernando *et al.*, 2016] are examples of problems within the area of recommendation that relies on representation vectors with intrinsic value. Their effectiveness can only be significantly improved if the intrinsic quality of the embeddings is adequately assessed. Therefore, evaluating the intrinsic properties of matrix factorization and neural embedding models is essential for ensuring their successful application across diverse contexts.

Evaluating the intrinsic quality of item embeddings is especially beneficial for methods that rely solely on user-item interactions during training and do not require item metadata. Generating intrinsically valuable representations without consuming intrinsic content is advantageous in scenarios where content data is scarce. However, no comprehensive study has focused on the intrinsic evaluation of item embeddings for CF recommender systems, with a few studies superficially addressing this topic.

To address this gap, we propose several methods for utilizing item metadata to intrinsically evaluate the vector representations of items within a CF recommender system. We introduce a widely used evaluation technique from recommender system literature, *similarity tables*, and adapt an intrinsic evaluation task from NLP to suit our context, *intruder detection*. Given the time-consuming and expertise-dependent nature of subjective analyses, we also present a novel quantitative, non-subjective strategy that leverages content-based data to assess the intrinsic ranking quality of embeddings, *content-based ranking comparison*, along with a task commonly explored in recommender systems, *automatic feature prediction*.

To motivate the execution of our evaluation framework, we complement the intrinsic evaluation with an extrinsic assessment of the vector representations' ability to generate high-quality recommendations. By comparing the results of these intrinsic and extrinsic evaluations, we reveal that embedding-based models can perform significantly differently across tasks. Notably, some models that underperform in recommendation tasks excel in intrinsic evaluations, emphasizing the importance of a comprehensive analysis when developing new representation models.

With these considerations, the primary objectives of this study are:

O1 Introduce new methods for evaluating item embeddings using well-known strategies derived from NLP, such as *intruder detection*, and from content-based recommendation, such as *content-based ranking comparison*;

O2 Present a collection of techniques for intrinsically evaluating item embeddings in both subjective and objective manners;

O3 Compare traditional embedding-based recommender models in extrinsic and intrinsic tasks to illustrate the varied performance of embeddings across different applications.

This paper builds on our previous work [Pires *et al.*, 2024], which introduced the preliminary framework for evaluating the intrinsic quality of item embeddings in recommender systems. In this extended version, we significantly expand on the original study by providing a more detailed explanation of the evaluation tasks, complemented by illustrations to enhance clarity and understanding. We also included novel metrics and examples and benchmarked a state-of-the-art model in both extrinsic and intrinsic tasks. Moreover, we highlight some potential problems and challenges when addressing intrinsic quality. These extensions offer a more comprehensive analysis, strengthen our conclusions, and aid readers in future research.

The study is organized as follows. Section 2 reviews the main related work. Section 3 outlines the research questions and goals. Section 4 offers two subjective approaches to evaluate the intrinsic quality of embeddings: one commonly used task and one adapted from NLP. Given the potential biases and time demands of subjective approaches, Section 5 presents two objective strategies, including a novel content-based ranking comparison. Section 6 describes the experimental setup, with results and analysis presented in Sections 7 and 8 for extrinsic and intrinsic evaluations, respectively. Finally, Section 9 provides conclusions and guidelines for future work.

2 Related work

The earliest collaborative filtering recommender systems relied on neighborhood-based methods to generate recommendations. However, as the number of users and items increased, these techniques faced significant challenges related to sparsity and scalability [Khsuro *et al.*, 2016]. As a result, the use of embeddings – low-dimensional vector representations that capture intrinsic meaning [Bengio *et al.*, 2003] – became increasingly popular. Initially, matrix factorization models were employed for this purpose [Koren *et al.*, 2009], reducing both memory consumption and processing demands while providing promising results. Over time, a variety of methods have been proposed to generate embeddings, incorporating diverse strategies such as graph-based algorithms and quantization techniques [Zhao *et al.*, 2023].

Inspired by state-of-the-art Natural Language Processing (NLP) techniques, recent methods aim to address sparsity and dimensionality issues by learning neural embeddings for items and users. Neural embeddings are dense, low-dimensional vector representations that carry intrinsic meaning and are learned using artificial neural networks [Ozsoy, 2016; Mikolov *et al.*, 2013]. Grbovic *et al.* [2015] presented the first approach using neural embeddings for recommender systems. The authors introduced Prod2Vec, a neural network for learning item embeddings similar to the Skip-gram architecture Mikolov *et al.* [2013], and User2Vec, a network capable of learning embeddings for users and items, inspired by Paragraph Vector Le and Mikolov [2014]. Both methods were used in an item recommendation problem based on receipts sent by e-mail and achieved interesting results compared to existing baselines.

In the following year, Barkan and Koenigstein [2016] proposed the Item2Vec, a neural network similar to Prod2Vec and Skip-gram, but ignoring the chronological ordering of interactions. In their study, the model achieved results superior to a singular value decomposition model. User2Vec and Item2Vec demand only the user-item interactions for learning the distributed vector representation without requiring ratings or metadata. Subsequent studies have built upon these foundational models, incorporating various techniques to enhance performance, e.g., consuming item metadata [Vasile *et al.*, 2016; FU *et al.*, 2017], leveraging content information to enrich embeddings [Zhang *et al.*, 2016; Greenstein-Messica *et al.*, 2017; Wang *et al.*, 2021].

Beyond incorporating item content, more complex neural models have been employed, including deep learning [Zarzour *et al.*, 2019; Sidana *et al.*, 2021], recurrent neural networks [Hidasi *et al.*, 2016; Wang *et al.*, 2021], convolutional neural networks [Tang and Wang, 2018; Ding *et al.*, 2023], GANs [Wang *et al.*, 2019; Chen *et al.*, 2022], variational autoencoders [Shenbin *et al.*, 2020], and transformers [Yu *et al.*, 2024; Kim *et al.*, 2024]. Another notable approach is training NLP models on textual data from items, users, or interactions [Siswanto *et al.*, 2018; Hasanzadeh *et al.*, 2020], with Large Language Models (LLMs) gaining attention in recent years [Gao *et al.*, 2023; Liu *et al.*, 2023].

A particularly intriguing aspect of neural embeddings is their ability to capture intrinsic meaning [Gladkova and Drozd, 2016]. However, few studies have thoroughly explored this property. The most commonly used approach for evaluating item embeddings involves similarity tables [Barkan and Koenigstein, 2016; FU *et al.*, 2017; Grbovic *et al.*, 2015], but this method heavily depends on subjective evaluations, which can be misleading or biased. Other techniques, such as genre plotting [Barkan and Koenigstein, 2016; FU *et al.*, 2017; Siswanto *et al.*, 2018] and sample clustering [Wang *et al.*, 2021; Kim *et al.*, 2024], also depend on human judgment or are difficult to apply across different domains, such as analogy analysis [Greenstein-Messica *et al.*, 2017].

In NLP, several established methods exist for intrinsic evaluation of word embeddings [Baroni *et al.*, 2014; Gladkova and Drozd, 2016; Qiu *et al.*, 2018; Yang *et al.*, 2022], including: (i) comparing human judgment of word similarity with embedding space similarity; (ii) predicting word analogies through vector arithmetic; (iii) clustering word embeddings and evaluating cluster quality; and (iv) detecting synonyms or "intruders" in groups of similar embeddings. However, many of these methods are challenging to adapt to recommender systems due to their reliance on external semantic datasets.

In many areas of recommender systems, the accuracy of the recommendation is prioritized over other characteristics such as quality [Werneck *et al.*, 2020], and although item embeddings are often measured in downstream applications such as recommendations, this does not guarantee intrinsic quality [Schnabel *et al.*, 2015]. High-quality embeddings can boost various tasks in a recommendation scenario, such as automatic feature prediction, user and item clustering, and knowledge discovery [Lu *et al.*, 2015; Hernando *et al.*, 2016], or enhance semantic-based recommendation engines, that use intrinsic knowledge as the main strategy for filtering items [Júnior and Manzato, 2015]. Embeddings with strong intrinsic value can even assist in uncovering item metadata for systems relying on categorical features [Wang and Lv, 2020; Chang *et al.*, 2023], which are often incomplete or inaccurate, and enhance traditional collaborative filtering models [Musto *et al.*, 2017].

To the best of our knowledge, no work on the recommender system literature has focused on studying the intrinsic aspects of item embeddings. Studies proposing novel embedding-based models often perform simple forms of intrinsic evaluation, normally based on subjective approaches that can add human bias and problems to the conclusions. Neglecting or improperly conducting intrinsic evaluations can result in the loss of valuable information that could support related tasks and improve representation models.

In this context, we propose alternative methods for the intrinsic evaluation of recommender system embeddings. In addition to presenting commonly used evaluation techniques, we adapt an NLP evaluation task to the recommender domain and introduce a novel quantitative metric for assessing intrinsic quality through a content-based ranking comparison. We conduct both extrinsic and intrinsic evaluations, comparing the results across different tasks to illustrate the diverse performance of embedding models.

3 Research questions and objectives

This study aims to inspire future research on distributed vector representations for items in Collaborative Filtering recommender systems by analyzing the intrinsic value of these representations alongside their performance in the recommendation task. Examining representation quality in extrinsic and intrinsic tasks can yield valuable insights into the model's capabilities and applications. With this goal in mind, we address the following three primary research questions:

RQ1 How we can evaluate the intrinsic quality of a representation model?

RQ2 How does a representation model's performance vary between intrinsic and extrinsic tasks?

RQ3 Are there significant differences in the outcomes of intrinsic evaluations that require human subjective opinions versus those that are calculated automatically?

To answer these research questions, we propose various strategies for intrinsically evaluating item embeddings using subjective and objective approaches. We then conduct experiments across multiple datasets to compare the performance of established models in both extrinsic and intrinsic evaluation frameworks. Finally, we analyze the results for each model, highlighting the key differences among the various evaluation tasks.

4 Subjective methods for intrinsic evaluation

Intrinsic evaluation of embeddings in both NLP and recommender systems frequently depends on subjective methods, relying on human judgment to assess the quality of the representations. In this section, we present two methods for evaluating the intrinsic quality of embeddings using subjective tasks. First, we describe similarity tables, one of the most common subjective approaches in the recommender system literature. Then, we introduce the intruder detection task, an evaluation scheme commonly used for NLP models that, even easily adaptable to recommender systems, is not well-explored in the field.

4.1 Similarity tables

The similarity table evaluation strategy involves training the compared models on the same datasets and selecting a known item as a seed. Next, the similarity between the embedding of the seed item A and the embeddings B for every other item $i \in I$ are calculated for each representation, traditionally using a measure of angular similarities, such as the cosine similarity (Equation 1). With the similarities calculated, the top-N nearest items for the seed, using each representation, can be retrieved and displayed side-by-side in a table.

$$cos(A,B) = \frac{A \cdot B}{||A||\,||B||} = \frac{\sum_{j=1}^{n}(A_j B_j)}{\sqrt{\sum_{j=1}^{n} A_j^2}\sqrt{\sum_{j=1}^{n} B_j^2}} \quad (1)$$

By addressing the item table, human evaluators can subjectively assess each group of nearest items to determine how well they match the seed item. This task can be repeated with multiple known items, providing a broader perspective on the representations' behavior. An illustration of this evaluation method is shown in Figure 1.

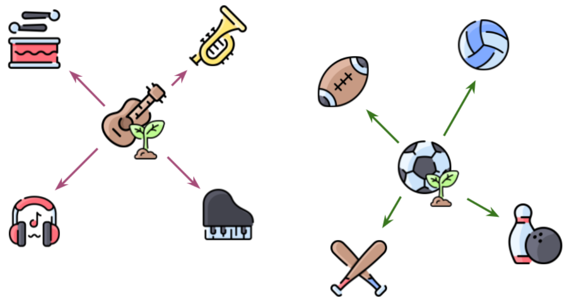

Figure 1. Construction of a similarity table. Seed items are selected and similar items in the vector space are retrieved.

Similarity tables are one of the most straightforward and intuitive ways to check for intrinsic meaning in the evaluated representations. This ease of use makes it one of the most commonly employed schemes [Barkan and Koenigstein, 2016; FU *et al.*, 2017; Grbovic *et al.*, 2015]. However, it heavily depends on human interpretation, as it measures intrinsic quality based on subjective analysis [Gladkova and Drozd, 2016].

4.2 Intruder detection

In NLP, intruder detection, also referred to as outlier detection consists of identifying "intruder" words within a set of neighbor words [Schnabel *et al.*, 2015]. We propose adapting this task for the recommender system domain by treating items as analogous to words.

The process begins with the selection of a few seed items, which can be randomly drawn or, preferably, curated by human evaluators. By retrieving the seed item embedding A and comparing with the remaining item embeddings B, it is possible to calculate a similarity neighborhood using metrics such as cosine similarity (Equation 1). Sets of items consisting of the seed item and its close neighbors are them constructed, which are expected to share semantic similarities. Later, a random item from the representation space is added to each set, as illustrated in Figure 2. Human evaluators are then tasked with identifying the "intruder" item (i.e., the randomly added one). The accuracy of their selections serves as a metric for comparing the quality of different embedding methods.

When selecting the seed items, the ones curated by human evaluators are likely to result in more homogeneous sets, making the task of finding the intruder easy. This usually happen because popular items tend to have more interactions, thus being more present during the training phase of the models and yielding better embeddings. Additionally, since they are well-known items, users can draw from past experience with the items to find the intruder, not necessarily relying on item metadata. However, since randomly-chosen seed items may be unfamiliar to most evaluators, they can highlight the ability of the embedding-based model to build well-defined sets for item that lack interactions.

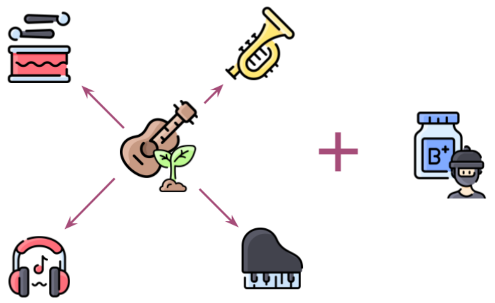

Figure 2. Intruder detection task. Seed items are selected, similar items in the vector space are retrieved and a random "intruder" item is sampled, to be discovered by human participants.

Although the task's outcome is quantitative, subjectivity is reduced but not eliminated. It still relies on human interpretation and can be time- and cost-intensive, requiring one or more individuals to analyze each chosen example. Many parameters or properties can also influence the analysis, such as the adopted neighborhood's size, the selected examples, and the fact that some items may have more similar neighbors than others [Gladkova and Drozd, 2016].

This approach is especially promissory in scenarios where researchers cannot access high-quality item metadata since most objective intrinsic evaluation depends on that type of information. In addition, some of the flaws of intruder detection, such as the demand for multiple evaluators, can be alleviated using crowdsourcing platforms, e.g., Amazon Mechanical Turk [1] and Prolific [2], which allows researchers to hire participants from diverse demographics.

[1]Amazon Web Services. *Amazon Mechanical Turk*. Available at `https://www.mturk.com/`.

[2]Prolific. *Prolific: Definitive human data to deliver world-leading research and AI*. Available at: `https://www.prolific.com/`

4.3 Shortcomings of subjective evaluation

A key limitation of the aforementioned subjective strategies lies in their heavy reliance on human judgment, which can often be biased or lack sufficient depth for specific domains [Faruqui *et al.*, 2016]. Human opinions may unfairly penalize embeddings when the grouping criteria determined by the algorithm diverge from those preferred by the evaluators [Senel *et al.*, 2018].

To avoid human bias, the experiments must be conducted with a vast range of people, which is more time-demanding and will most likely result in cost increases. To overcome subjectivity in the existing evaluation schemes, it is recommended to use objective approaches, as introduced in the following section.

5 Objective methods for intrinsic evaluation

As mentioned, the weakest point of using subjective strategies resides in their high dependence on human opinions. To overcome subjectivity in the existing evaluation schemes, it is recommended to use objective approaches, i.e., the ones that consume additional data to quantify the intrinsic quality of the embeddings and are calculated automatically. Here, we suggest the use of two different metrics. First, we describe the task of automatic feature prediction, a well-known problem in recommender systems. Lastly, we present a content-based ranking comparison approach using the Normalized Discounted Cumulative Gain as an example metric.

5.1 Automatic feature prediction

Automatically discovering item features, often referred to as auto-tagging, involves predicting a set of unknown tags for a target item i based on its most similar items. This task represents a specialized problem in the domains of recommender systems and knowledge discovery, with numerous methods developed specifically to address it. [Song *et al.*, 2011; de Souza P. Moreira *et al.*, 2019; Lisena *et al.*, 2022].

We can employ a neighborhood-based automatic feature prediction approach to assess the intrinsic value of a vector representation [Barkan and Koenigstein, 2016; FU *et al.*, 2017]. For each item i in the entire catalog, its k nearest items are selected, considering the embedding-vector space. Next, the attributes of the target item are predicted through some voting process, such as a simple majority vote, in a similar way as illustrated in Figure 3. We can then compare the original item's attributes with the ones predicted by the neighbors and quantify the intrinsic quality of the representation using traditional classification metrics, such as precision, recall, or F1-score.

Although predicting features may not be entirely trustworthy since there may be some noise and false information (especially if they are user-informed), in most cases, this remains the only available source of information that captures the intrinsic content of an item.

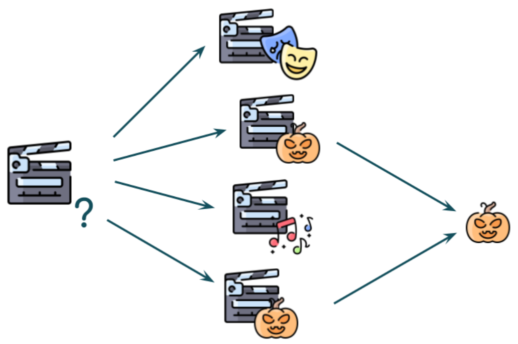

Figure 3. Automatic feature prediction task. Random items are sampled and their features are estimated according to their neighborhood on the vector space.

5.2 Content-based ranking comparison

As an additional objective and automatic method for intrinsically evaluating vector representations of items, we propose a novel metric that compares the neighborhood generated by the embeddings in the vector space with a neighborhood constructed using content-based information about the items, as illustrated by Figure 4. The quality of the comparison can then be quantified using a ranking comparison metric.

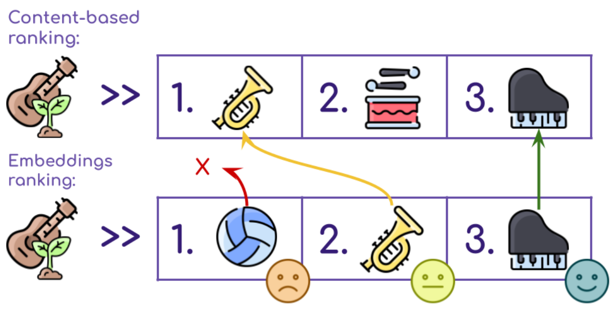

Figure 4. Content-based ranking comparison. Seed items are selected, and their most similar items in content-based representation are ranked and treated as target items. Embedding-based ranking is then generated and compared with the content-based ranking, yielding better scores when there is a closer match in both selected items and ranking position.

To properly evaluate the spatial distribution generated by the learned embeddings, we first assume that there is a correct order for the neighborhood of a given item when filtering its most similar items on the embeddings vector space. In this study, we constructed the target ordering using the similarities between the high-level features of the items: the item's category, genre, or tags. Although this strategy is commonly used in traditional content-based recommender systems, it is important to clarify that this may not be the proper neighborhood when recommending items. Many studies show that there are better and more complex content-based approaches that yield better recommendations [Pazzani and Billsus, 2007]. As we aim to approximate items with similar features, we can compare this neighborhood with the one generated by the embeddings. Moreover, it is possible to use more complex and domain-specific techniques for ranking, e.g., low-level visual features for movies [Filho *et al.*, 2017] and context and metadata graph embeddings for music [Wang *et al.*, 2017].

Using those general features to describe the items, we can represent an item i through a bag-of-words encoding, i.e., an array of attributes $\vec{i}$ according to Equation 2. These attributes describe the item's characteristics, which can vary according to the recommendation domain. For instance, we may use information like actors, directors, and genre when recommending movies. We can then represent every item as a vector with size according to the number of attributes.

$$\vec{i}_a = \begin{cases} 1, & \text{if the item } i \text{ has the attribute } a \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

With this representation, we can build a similarity matrix $\mathcal{C}$ of dimensions $|I| \times |I|$, in which $|I|$ represents the number of items in the catalog. Thus, for a given pair of items i and j, $\mathcal{C}_{i,j}$ stores their similarity when using the content-based representation, i.e., vectors $\vec{i}$ and $\vec{j}$, calculated using a metric such as cosine similarity. Similarly, we construct the similarity matrix $\mathcal{E}$, which stores the similarity of the items' dense embeddings for every pair of items i and j.

Afterward, for each item $i \in I$, we can construct two neighborhoods, $\mathcal{N}_i^{\mathcal{C}}$ and $\mathcal{N}_i^{\mathcal{E}}$. The former corresponds to the subset of items most similar to i considering the similarity matrix $\mathcal{C}$, i.e., the items that share the most related content-based features. The latter represents the same neighborhood concept but uses the similarity values stored in matrix $\mathcal{E}$. Both $\mathcal{N}_i^{\mathcal{C}}$ and $\mathcal{N}_i^{\mathcal{E}}$ may be limited to a restricted number of neighbors, defined by a hyperparameter k, to reduce memory consumption when calculating the ranking.

With both similarity matrices constructed, we can compare them using different metrics and approaches, e.g., traditional or utility-based ranking measures, and sample sets' similarity metrics. Regardless of the adopted strategy, all of them are maximized according to the same ordering, i.e., the one created using the content-based representation, with the differences being in how the neighborhoods are used to calculate the final score. In the following, we offer metrics for each one of those approaches.

5.2.1 Rank correlation metrics

When you have sorted data and a reference ranking, in that case the content-based one, it is possible to compare the rankings using metrics of rank correlation such as Spearman's rank correlation coefficient ρ, Kendall's τ coefficient, or the Normalized Distance-based Performance Measure (NDPM).

For comparing the rankings using Spearman's ρ, we can calculate the correlation coefficient for each item and average the results, as shown in Equation 3:

$$\rho = \frac{\sum_{i \in I} \rho_i}{|I|} \tag{3}$$

in which ρ_i represents the rank correlation coefficient for a specific item i. To calculate ρ_i, we must first determine each item's rank value in $\mathcal{N}_i^{\mathcal{C}}$ and $\mathcal{N}_i^{\mathcal{E}}$. In cases where multiple items share the same similarity, the same average rank is used. Then, we can calculate ρ_i as shown in Equation 4:

$$\rho_i = 1 - \frac{6 \sum_{j \in I} d_j^2}{|I|(|I|^2 - 1)} \tag{4}$$

in which d corresponds to the difference among ranking positions. That is, considering that, for a given item j, $\text{pos}_j^{\mathcal{C}}$ corresponds to the ranking position of j when sorting $\mathcal{N}_i^{\mathcal{C}}$, and $\text{pos}_j^{\mathcal{E}}$ when sorting $\mathcal{N}_i^{\mathcal{E}}$, d_j is given by Equation 5:

$$d_j = \text{pos}_j^{\mathcal{C}} - \text{pos}_j^{\mathcal{E}} \tag{5}$$

5.2.2 Set similarity metrics

If we limit the neighborhoods to the top-K similar items, we can treat them both as sample sets and calculate metrics designed to compare the similarity and diversity of sets, such as the Jaccard Index or the Sørensen–Dice coefficient.

For the Jaccard Index J, we must first calculate the Jaccard Index J_i for each item i, and then average the results, as shown in Equation 6. For J_i, we must build two sets, $S_i^{\mathcal{C}}$ and $S_i^{\mathcal{E}}$, consisting of the top-K most similar items from both $\mathcal{N}_i^{\mathcal{C}}$ and $\mathcal{N}_i^{\mathcal{E}}$, respectively, and then calculate the set similarity using Equation 7:

$$J = \frac{\sum_{i \in I} J_i}{|I|} \tag{6}$$

$$J_i = \frac{|S_i^{\mathcal{C}} \cap S_i^{\mathcal{E}}|}{|S_i^{\mathcal{C}} \cup S_i^{\mathcal{E}}|} \tag{7}$$

5.2.3 Utility-based ranking scores

Finally, we can also adapt utility-based ranking metrics, using the content-based similarities to quantify the utility of the embeddings ranking.

Here, we explain how the Normalized Discounted Cumulative Gain (NDCG) can be adapted for this type of evaluation. First, as it is commonly calculated for the metric, we define the overall NDCG as the average of the NDCG for each item i (NDCG$_i$), as shown in Equation 8. NDCG$_i$, on the other hand, is defined as the Discounted Cumulative Gain of the item (DCG$_i$) divided by its Ideal Discounted Cumulative Gain (CB-IDCG$_i$), normalizing the final value to a 0–1 range, as shown in Equation 9.

$$\text{NDCG} = \frac{1}{|I|} \sum_{i \in I} (\text{NDCG}_i) \tag{8}$$

$$\text{NDCG}_i = \frac{\text{DCG}_i}{\text{IDCG}_i} \tag{9}$$

The main differences arise when calculating the DCG$_i$ and the IDCG$_i$. For both scores, we consider that the "gain" of a given item j in the neighborhood of i is given by the values of matrix $\mathcal{C}$, i.e., the content-based similarity matrix. The ideal gain, IDCG$_i$, is retrieved using the top-k items of the content-based neighborhood, $\mathcal{N}_i^{\mathcal{C}}$, while the obtained gain, DCG$_i$ uses the embedding-based neighborhood, $\mathcal{N}_i^{\mathcal{E}}$, as presented in Equations 10 and 11, respectively.

$$\text{IDCG}_i = \sum_{n=1}^{k} \frac{C_{i,\mathcal{N}_{i_n}^{\mathcal{C}}}}{\log_2(n+1)} \tag{10}$$

$$\text{DCG}_i = \sum_{n=1}^{k} \frac{C_{i,\mathcal{N}_{i_n}^{\mathcal{E}}}}{\log_2(n+1)} \tag{11}$$

For the ideal score (Equation 10), we defined the gain for the n_{th} item in the neighborhood as $C_{i,\mathcal{N}^{\mathcal{C}}_{i_n}}$, which corresponds to the content-based similarity stored in $\mathcal{C}$ between the target item i and the n_{th} item of i's neighborhood in $\mathcal{N}^{\mathcal{C}}$. For the generated score (Equation 11), the gain is calculated similarly. It is represented by $C_{i,\mathcal{N}^{\mathcal{E}}_{i_n}}$, corresponding to the similarity stored in $\mathcal{C}$, but between the target item i and the n_{th} item of i's neighborhood in $\mathcal{N}^{\mathcal{E}}$.

As we use the content-based similarity to weigh the final result, the higher the final NDCG, the more the neighborhood of the embeddings matches the neighborhood by content, both in the choice of items and their order.

5.2.4 General problems when comparing content-based rankings

It is important to highlight that when using high-level features with few variations, e.g., categories or genres, it is common for many items to be represented as the same feature vector. In that case, it may happen that many items will share the same similarity score with the target item. This may not be a problem in full-ranking measures that deal with tied rankings, such as Spearman's ρ. However, this can lead to problems for the Jaccard Index and NDCG, which selects a subset of neighboring items. In our experiments, if tied items were in the threshold k, we randomly drew the neighborhood items from $\mathcal{N}^{\mathcal{C}}_i$. The same problem is unlikely to happen with $\mathcal{N}^{\mathcal{E}}_i$ given that the similarities are computed using dense vectors of continuous values. Even so, for the NDCG, since we compare the neighborhoods using the similarity in $\mathcal{C}$, if a subset of items shares the same similarity in $\mathcal{N}^{\mathcal{C}}_i$, the ordering between them in $\mathcal{N}^{\mathcal{E}}_i$ will not impact the final score.

The main advantage of a ranking-comparison method is that it is calculated automatically, eliminating human interference and subjectivity. It also assesses the ordering quality given by the representations beyond the neighborhood. As a con, the main weakness of the measure is that it depends heavily on the hypothesis that content-based ordering is a good indicator of intrinsic similarity, which may not be accurate in all scenarios. Even so, this assumption is used in most content-based recommender problems and can provide insights into the intrinsic quality of the representation [Pazzani and Billsus, 2007].

6 Experimental setup

Although the main objective of this paper is to introduce a guideline for assessing the intrinsic quality of item embeddings, we also performed the aforementioned strategies for a series of established baseline methods over multiple well-known benchmark datasets. Besides being a practical example of conducting the proposed pipeline, the presented results highlight the importance of performing an intrinsic evaluation.

This section details the experimental setup. First, we present the datasets used in the experiments, then we describe the benchmark algorithms and the fine-tuning phase, followed by an explanation of the adopted strategy for hyperparameter tuning.

To give credibility to the results and make the experiment reproducible, the source code is publicly available[3].

6.1 Datasets and data preprocessing

Table 1 presents the datasets used in the experiments. They are publicly available and have been extensively used in the literature or challenges. Information about the repositories and websites where the datasets are stored can be found in the *Declarations* section at the end of the article.

To properly evaluate the ability of the recommender algorithms to generate high-quality recommendations, we selected multiple datasets to conduct the extrinsic evaluation. Not every dataset provides item metadata, thus the intrinsic evaluation was conducted over only a portion of the datasets, as listed in column "Intrinsic" in Table 1.

For datasets that contain item metadata, the features describing the items can be of two types:

- Categories: attributes inherent to the item, informed by the system owner and more general to the item, e.g. movie genre or product characteristics;
- Tags: values informed by users without moderation, which can result in many different values, generally more specific for the item.

Since user-informed tags are liable to noise and inconsistency, we opt to use only the top-100 most recurring tags by dataset, as performed in studies of tag-based recommender systems [Eck *et al.*, 2007; Firan *et al.*, 2007]. For the ML-25m dataset, we have used only the categories, since it is a more properly curated information.

6.2 Embedding-based models

Although the area is facing a rapid and promising increase in deep learning models for generating item and user embeddings, recent studies have shown that matrix factorization methods can still outperform more complex models in many traditional collaborative filtering benchmarks, with the advantage of being computationally efficient [Rendle *et al.*, 2020, 2022]. With that in mind, we have implemented two well-known methods for matrix factorization, Alternating Least Squares (ALS) [Hu *et al.*, 2008] and Bayesian Personalized Ranking (BPR) [Rendle *et al.*, 2009], using the `implicit` [4] library. Moreover, considering that a good intrinsic meaning is commonly achieved by context-window models [Mikolov *et al.*, 2013], we have also implemented two contextual neural embeddings-based recommenders, Item2Vec (I2V) [Barkan and Koenigstein, 2016] and User2Vec (U2V) [Grbovic *et al.*, 2015], using the `gensim` [Řehůřek and Sojka, 2010] library. Lastly, to compare the selected models with a state-of-the-art deep learning model, we executed RecVAE (RVA) [Shenbin *et al.*, 2020], a variational autoencoder for implicit recommendation, using

[3] Source code of the experiments: `https://github.com/UFSCar-LaSID/recsys-intrinsic-evaluation`

[4] Ben Frederickson. 2017. *Implicit: Fast Python Collaborative Filtering for Implicit Datasets*. Available at: `https://github.com/benfred/implicit`

Intrinsic	Dataset	Users	Items	Interactions	Sparsity	Domain	Categories	Tags
✓	**Anime**	73,514	11,200	7,813,733	99.05%	Movies	43	
✓	**BestBuy**	1,268,702	69,858	1,865,269	99.99%	Retail	1,540	
	BookCross	59,517	246,724	716,109	99.99%	Books		
	CiaoDVD	17,615	16,121	72,345	99.97%	Movies		
✓	**Delicious**	1,867	69,223	104,799	99.92%	Websites		14,346
	FilmTrust	1,508	2,071	35,494	98.86%	Movies		
✓	**Last.FM**	1,892	17,632	92,834	99.72%	Music		9,718
✓	**ML-25m**	162,541	59,047	25,000,095	99.74%	Movies	20	65,464
	NetflixPrize	480,189	17,770	100,480,507	98.82%	Movies		
	RRocket	11,719	12,025	21,270	99.98%	Retail		

Table 1. Description of each dataset used in the experiments. Blank spaces are used for datasets without categories or tags.

its original implementation[5]. All models were selected based on their popularity in recent studies (number of citations) and ease of replication.

For fairness, we only compare methods that do not rely on item metadata during the learning phase since we aim to evaluate the representations' ability to learn the intrinsic value of items. Any intrinsic value is learned using only data from user-item interactions, showing the power that those methods can have for figuring out knowledge about the items without consuming this information. They also have the advantage of being easily applicable in most scenarios.

6.3 Hyperparameter optimization

We performed a hyperparameter optimization through a grid search for each method and dataset. For this, we have used holdout, with 80% of the interactions for training, 10% for validation, and 10% for testing. The main objective was to maximize NDCG in a top-15 recommendation [Shani and Gunawardana, 2011]. Due to the limited resources available to train the models, which are very time and processing-consuming, we have randomly selected only 10% of the interactions from ML-25m dataset to optimize the hyperparameters. For the remaining datasets, we used all interactions.

For ALS and BPR, we tested latent factors of different sizes $f = \{50, 100, 300\}$, regularization factor $\lambda = \{0.01, 0.1, 1\}$ and learning rate $\alpha = \{0.0025, 0.025, 0.25\}$, using 100 epochs for training. For the context-window models, we varied the number of epochs $n = \{50, 100, 200\}$, subsampling rate of frequent items $t = \{10^{-5}, 10^{-4}, 10^{-3}\}$, and exponent to shape the negative sampling distribution $\alpha = \{-1.0, -0.5, 0.5, 1.0\}$, as recommended by Caselles-Duprés *et al.* [2018]. For the RecVAE model, we ranged γ in $\{0.0035, 0.005, 0.01\}$ and the remaining hyperparameters fixed to the values recommended by its authors [Shenbin *et al.*, 2020]. For any unmentioned parameter, we used the library's default values.

7 Extrinsic results

To assess the representation models' ability to provide good recommendations, i.e., to evaluate the models extrinsically, we conducted a top-N ranking task, with N varying in $\{10, 15, 20\}$. After learning the vector representation of each model, we generated the recommendations following the original strategy proposed by each algorithm. We then calculated the F1-score and the Normalized Discounted Cumulative Gain (NDCG), presented in Tables 2 and 3, respectively. Considering that the results were similar for multiple values of N, we opt to present only the results for $N = 15$. In both tables, the darker-toned cells, the better the results. The best overall results for each combination of dataset and the value for N are highlighted in **bold**.

Dataset	**Representation Model**				
	ALS	**BPR**	**I2V**	**U2V**	**RVA**
Anime	0.134	0.090	0.088	0.007	**0.156**
BestBuy	0.014	0.016	0.017	0.005	**0.036**
BookCross	**0.010**	0.006	0.008	0.001	0.009
CiaoDVD	0.015	0.012	0.012	0.003	**0.022**
Delicious	0.023	0.020	0.039	0.013	**0.691**
FilmTrust	0.110	0.241	0.278	0.127	**0.280**
Last.FM	0.094	0.078	0.105	0.016	**0.116**
ML-25m	0.175	0.110	0.098	0.000	**0.204**
NetflixPrize	0.042	0.031	0.054	0.001	**0.060**
RRocket	0.022	0.002	0.018	0.006	**0.029**

Table 2. F1@15 achieved by each algorithm in each dataset.

Dataset	**Representation Model**				
	ALS	**BPR**	**I2V**	**U2V**	**RVA**
Anime	0.237	0.174	0.128	0.008	**0.250**
BestBuy	0.063	0.075	0.056	0.016	**0.149**
BookCross	0.017	0.010	0.013	0.001	**0.018**
CiaoDVD	0.041	0.030	0.028	0.006	**0.056**
Delicious	0.055	0.047	0.097	0.024	**0.193**
FilmTrust	0.238	0.557	0.572	0.195	**0.616**
Last.FM	0.186	0.160	0.189	0.023	**0.205**
ML-25m	0.273	0.168	0.121	0.000	**0.321**
NetflixPrize	0.058	0.043	**0.064**	0.001	0.054
RRocket	0.123	0.002	0.083	0.013	**0.138**

Table 3. NDCG@15 achieved by each algorithm in each dataset.

The behavior of the models was similar when comparing both metrics, with the exception of BookCross and Netflix-

[5]RecVAE source code. Available at: `https://github.com/ilya-shenbin/RecVAE`.

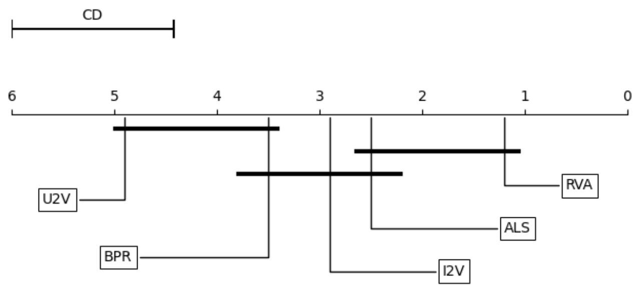

Figure 5. Visual representation of the Nemenyi test containing the critical difference (CD) and the models' average NDCG@N rank.

Prize datasets, in which ALS and Item2Vec achieved the best F1-Scores, respectively, while RecVAE achieved the best NDCG.

Results indicate a clear superiority of RecVAE. The SOTA model achieved the best results in 90% of the datasets, being the second or third-best in the few cases where it was surpassed by another algorithm. We can also see a similar performance between ALS, BPR, and Item2Vec, with each model's performance varying according to the datasets. User2Vec, on the contrary, was the worst embedding-based model, obtaining low results for every dataset and metric.

To properly analyze the models, we conducted a non-parametric Friedman test to verify if there is a statistically significant difference between them [Demšar, 2006], using a ranking constructed with the NDCG, since the metric addresses not only the correctness of the recommendation but also the ordering of items. The test indicates that the models differ, with 99% confidence ($X_F^2 = 26.04$). We then conducted a Nemenyi test to perform a pairwise comparison [Demšar, 2006]. A critical difference (CD) of 1.575 was obtained using Equation 12, and the distance between the average rankings of each model was compared.

$$CD = q_\alpha \sqrt{\frac{a(a+1)}{6D}} \tag{12}$$

in which D corresponds to the number of datasets ($D = 10$), a corresponds to the number of algorithms ($a = 5$) and q_α is the adopted critical value, in this case, 2.728.

With 95% confidence, we can say that there is statistically significant evidence that RecVAE is superior to every method except for ALS, considering that the differences in the average rankings were superior to the critical difference. We can also say that ALS and Item2Vec are statistically superior to User2Vec, which was the worst-ranked model among all five. A visual representation of the Nemenyi test is presented in Figure 5, with bars representing the CD.

Results of the extrinsic experiment show that RecVAE is consistently the superior model for yielding recommendations through the learning of user and item embeddings. Conversely, BPR is the most unsuitable method for this task compared to the others. However, following insights for the NLP area [Schnabel *et al.*, 2015], this behavior may not necessarily repeat when we apply the same representation vectors to intrinsic tasks. Therefore, we conducted different intrinsic evaluation strategies to assess this particularity, as described in the following section.

8 Intrinsic results

To intrinsically evaluate the vector representation, we performed the tasks presented in Sections 4 and 5, discussing the main differences of each experiment. The results show how the same representation model can perform differently according to the task, especially when comparing subjective to objective strategies.

8.1 Similarity tables

We built two similarity tables using popular items from datasets of widely known domains: Last.FM (Table 4) and ML-25m (Table 5), along with their top-3 neighbors in each representation. We used those datasets because they are from intuitive and widely known domains (music and movies). Due to space constraints, some artists' and movies' names have been shortened.

In Table 4, the methods generally found a neighborhood with similar items to the target. In most cases, the bands and artists have completely varied for each representation model, with only a few exceptions such as *Jay-Z*, *30 Seconds to Mars*, and *Beyoncé*, which were present on three of the algorithms. Even with the selection of different artists, the music genres were normally very related to the target. Some methods behave in a more conservative manner, such as BPR returning only hip-hop and rap artists for *Eminem*, while others returned relevant items, but deviating from the tags, such as *Ke$ha* and *P!nk* in the Item2Vec neighborhood for *Eminem*.

Contradicting the results obtained in the extrinsic evaluation, RecVAE was the worst model for selecting similar artists. In the neighborhood of *The Beatles*, a classic rock band from the 60s, the representation returned popular rock bands, but from different sub-styles and decades. For other seed items, the results were even more unexpected, such as the electronic musical project *Recoil* for rock n' roll star *Elvis Presley*, glam rock bassist and singer *Suzi Quatro* for rapper *Eminem*, and Brazilian rock band *Biquini Cavadão* for Colombian pop singer *Shakira*. ALS also presented some tag contradictions in its results, such as the learned neighborhood for *The Beatles*, which, instead of other rock or 60s bands, contains Arabic and baroque artists.

For Table 5, the algorithms in general tended to behave worse. BPR and User2Vec, especially, found some very related items, such as the sequels for *X-Men*. However, they also found some odd neighbors, such as *Jack-O* or *Men in Black*, a horror and sci-fi movie, respectively, for *Toy Story*, a children's animation. For some movies, such as *Titanic*, all methods performed poorly when comparing the genres between target and neighbor items. On the other hand, some of the neighboring movies are considered classic films, implying that the representations have discovered a pattern. Due to these conflicting results, it is hard to select a superior representation without relying on human personal opinions. When analyzing which algorithm had the most trouble in finding similar items, RecVAE was consistently the model that yielded the most unexpected items, such as *Just Go With It*, a light-hearted comedy, for *Friday the 13th*, a classic horror movie, and *Deadpool* and *Beavis and Butthead Do Amer-*

Seed item	**#**	**Representation Model**				
		ALS	**BPR**	**I2V**	**U2V**	**RVA**
The Beatles *classic rock, rock*	**1st**	Ricky Nelson *rock, classic rock*	The Beach Boys *60s, classic rock*	David Bowie *rock, classic rock*	The Kinks *60s, classic rock*	Interpol *indie, alternative*
	2nd	Souad Massi *female, arabic*	John Lennon *classic rock, rock*	Radiohead *alternative, rock*	Rolling Stones *classic rock, rock*	The Killers *indie, indie rock*
	3rd	Andrés Segovia *baroque, guitar*	Ringo Starr *classic rock, rock*	Led Zeppelin *hard rock, rock*	Velvet Underground *psychedelic, rock*	Muse *prog rock, indie*
Elvis Presley *classic rock, rock*	**1st**	Lauren Nichols *female, country*	Bruce Springsteen *rock, classic rock*	John Lennon *classic rock, rock*	Paul McCartney *classic rock, rock*	Recoil *electronic, trip-hop*
	2nd	The Merseybeats *classic rock, british*	The Mamas & The... *classic rock, 60s*	The Beatles *classic rock, rock*	Lacrimosa *gothic, metal*	Justin Timberlake *pop, rnb*
	3rd	Steve Gaines *classic rock, 70s*	Elton John *pop, classic rock*	The Doors *classic rock, rock*	Eddie Cochran *rockabilly, rock*	George Harrison *classic rock, rock*
Eminem *hip-hop, rap*	**1st**	Ice Cube *hip-hop, rap*	Jay-Z *hip-hop, rap*	Ke$ha *pop, dance*	Akon *hip-hop, rap*	Floetry *soul, rnb*
	2nd	Bizarre *hip-hop, rap*	50cent *hip-hop, rap*	P!nk *pop, female*	Nelly *hip-hop, rap*	Suzi Quatro *glam rock, hard rock*
	3rd	Xzibit *hip-hop, rap*	Kanye West *hip-hop, rap*	Jay-Z *hip-hop, rap*	Jason Derulo *pop, rnb*	Jay-Z *hip-hop, rap*
Linkin Park *rock, nu metal*	**1st**	medusa'scream *emo*	30 Seconds to Mars *alt rock, rock*	Breaking Benjamin *alt rock, rock*	30 Seconds to Mars *alt rock, rock*	Guano Apes *alt rock, rock*
	2nd	Grey Daze *alt rock, rock*	The Rasmus *rock, alternative*	30 Seconds to Mars *alt rock, rock*	Flyleaf *alt rock, rock*	Breaking Benjamin *alt rock, rock*
	3rd	Dead By Sunrise *alt rock, rock*	Breaking Benjamin *alt rock, rock*	Evanescence *rock, female*	VersaEmerge *rock, female*	Rammstein *metal, industrial*
Shakira *female, pop*	**1st**	Juanes *latin, pop*	Beyoncé *rnb, pop*	Rihanna *pop, rnb*	Katy Perry *pop, female*	Biquini Cavadão *brazil, rock*
	2nd	Fanny Lu *latin, pop*	Marilyn Monroe *jazz, female*	Beyoncé *rnb, pop*	Mariah Carey *rnb, pop*	Cher *pop, dance*
	3rd	Thalía *latin, pop*	Rihanna *pop, rnb*	Britney Spears *pop, dance*	Beyoncé *rnb, pop*	Jonas Brothers *pop rock, pop*

Table 4. Similarity table of five popular artists from the Last.FM dataset.

ica, both acid and dark humor comedies, for *Titanic*, a classical drama. The results are once again surprising when compared to the demonstrated superiority of the model on the extrinsic evaluation.

As mentioned, this evaluation method is heavily influenced by human subjectivity. For instance, in Table 4, among all neighborhoods of *Shakira*, the ones composed by *Rihanna*, *Britney Spears*, *Katy Perry*, *Mariah Carey* and *Beyoncé* would be the most similar if we consider that they are all world-famous pop artists. However, *Thalía*, *Fanny Lu*, and *Juanes* are all Latin pop artists; hence, they are strongly connected with the target, *Shakira*. Therefore, deciding what is more similar is complex, and our opinions and backgrounds can strongly skew our guesses.

8.2 Intruder detection

We conducted the task using Anime, ML-25m, and Last.FM datasets, as they are from well-known domains and contain well-curated additional information, e.g., genre and release year. For each dataset, we selected 15 items to use as seeds, of which 10 were popular items and 5 were completely random. We then built five questionnaires, each with 15 items, alternating between the representation models. Finally, we asked a group of 10 individuals to discover the intruder. A few examples of sets with intruders that were used in the interview are shown in Table 6. They are related to the ALS model trained over Last.FM dataset, using popular and random artists as seeds. Although the seed and intruder items are defined in the example, they were hidden and shuffled when presenting the questionnaires to the evaluators.

The intruder test was first performed with ALS, BPR, Item2Vec, and User2Vec [Pires *et al.*, 2024]. When the RecVAE was analyzed in this study, the same individuals could not participate again in the task. Therefore, we opted not to include the results obtained by RecVAE to allow a fair comparison between the models. Each of these representation models received 30 votes, and the accuracy for each of them is shown in Tables 7, 8 and 9. The tables consist, respectively, of the general accuracy, the accuracy for the 10 popular items, and the accuracy for only the 5 random (and probably unknown) items. All results are in grayscale; the darker the cell, the greater the accuracy.

BPR and User2Vec performed better at building a good quality neighborhood, as they usually presented the highest number of correct answers per dataset. Item2Vec also constructed a satisfactory neighborhood, being a close second in almost every case.

BPR generally achieved the best results in the scenario considering all items, being the best model for ML-25m and

Seed item	**#**	**Representation Model**				
		ALS	**BPR**	**I2V**	**U2V**	**RVA**
Friday the 13th *horror, thriller*	**1st**	A View to Kill *action*	Nigthmare in Elm... *horror*	Gremlins 2 *comedy, horror*	Friday the 13th II *horror*	Two Evil Eyes *horror*
	2nd	Child's Play *horror, thriller*	Friday the 13th III *horror*	Texas Chainsaw... *horror*	Halloween II *horror*	Just Go With It *comedy, romance*
	3rd	Pet Sematary *horror*	Children of the Corn *horror*	Halloween *horror*	Child's Play *horror, thriller*	Death Becomes Her *comedy, fantasy*
Grease *musical*	**1st**	Dirty Dancing *musical*	Sound of Music *musical*	Batman Returns *action, crime*	Top Gun *action*	People vs. Larry... *comedy, drama*
	2nd	Big *comedy*	A Little Princess *children*	Scream *horror*	Gremlins *comedy, horror*	Liar Liar *comedy*
	3rd	Little Mermaid *musical*	Oliver! *musical*	Air Force 1 *action, thriller*	Mary Poppins *musical*	Blood & Wine *crime, drama*
Titanic *drama*	**1st**	Groundhog Day *comedy*	Truman Show *comedy*	Good Will Hunting *drama*	Jurassic Park *action, sci-fi*	The Usual Suspects *crime, mistery*
	2nd	The Truman Show *comedy*	Catch Me If You Can *crime, drama*	Men in Black *action, sci-fi*	Truman Show *comedy*	Deadpool *action, comedy*
	3rd	Christmas Do-Over *comedy*	My Best Friend's... *comedy*	Saving Private Ryan *drama, war*	Men in Black *action, sci-fi*	Beavis and Butthe... *animation, comedy*
Toy Story *children*	**1st**	Average Italian *comedy*	Muppet Treasure... *children*	Braveheart *drama, war*	Lion King *children*	Beauty and the Beast *animation, children*
	2nd	The Pride & The P... *war, action*	Babe *children*	12 Monkeys *sci-fi, thriller*	Toy Story 2 *children*	Total Recall *action, sci-fi*
	3rd	Barbie *animation*	Jack-O *horror*	The Usual Suspects *crime*	Men in Black *action, sci-fi*	Waking Ned Devine *comedy*
X-Men *action, sci-fi*	**1st**	Star Wars II *sci-fi*	Monsters, Inc. *children*	Toy Story 2 *children*	Star Wars II *sci-fi*	Insomnia *thriller, mistery*
	2nd	The Matrix 2 *action, sci-fi*	X-Men United *action, sci-fi*	Gladiator *action*	X-Men United *action, sci-fi*	How to Lose a Guy... *comedy, romance*
	3rd	Star Wars I *sci-fi*	Spider-Man *adventure*	Memento *mystery*	Gladiator *action*	Spider-Man *action, adventure*

Table 5. Similarity table of five popular movies from the ML-25m dataset.

Seed criterion	**Seed artist**	**Neighbors**		**Intruder**
Popular	AC/DC *hard rock, classic rock*	Led Zepellin *classic rock, hard rock*	Ozzy Osbourne *hard rock, metal*	Delinquent Habits *hip-hop, rap*
	Bob Dylan *songwriter, classic rock*	The Band *classic rock, folk rock*	The Byrds *60s, classic rock*	Gong *psychedelic, prog rock*
Random	Reverend Horton *rock, favorites*	Halou *trip-hop, favorites*	Bitcrush *ambient, experimental*	Brown Eyed Girls *kpop, asian*
	David Usher *rock, alternative*	Reel Big Fish *punk, ska*	Ben Folds Five *alternative, piano*	Oceano *deathcore, instrumental*

Table 6. Examples used in the intruder detection task for ALS model and Last.FM dataset.

Dataset	**Representation Model**			
	ALS	**BPR**	**I2V**	**U2V**
Anime	43.33%	63.33%	60.00%	**90.00%**
ML-25m	33.33%	**66.67%**	46.67%	43.33%
Last.FM	66.67%	**90.00%**	86.67%	66.67%

Table 7. Accuracy for the intruder detection task (all items).

Dataset	**Representation Model**			
	ALS	**BPR**	**I2V**	**U2V**
Anime	44.44%	66.67%	55.56%	**83.33%**
ML-25m	38.89%	**61.11%**	44.44%	44.44%
Last.FM	72.22%	**94.44%**	88.89%	77.78%

Table 8. Accuracy for the intruder detection task (popular items).

Dataset	Representation Model			
	ALS	BPR	I2V	U2V
Anime	41.67%	58.33%	66.67%	**100.00%**
ML-25m	25.00%	**75.00%**	50.00%	41.67%
Last.FM	58.33%	**83.33%**	**83.33%**	50.00%

Table 9. Accuracy for the intruder detection task (random items).

Last.FM, and the second-best for Anime. User2Vec presented promising results for the Anime dataset and reached 100% accuracy in the scenario where items were randomly selected, showing that it could generate a relevant neighborhood even in cases where there is little knowledge about the item. This result is exciting, considering the scores obtained by the model in the extrinsic evaluation (Section 7).

The intruder detection approach seems more robust than the similarity matrix, as it generates a quantitative value. However, its weakness is that human influence is still considerably present. When asked about the test, the group of guessers reported difficulty finding some intruders and choosing at random in some cases. This can distort the results and confirms the need for less human-based evaluation techniques.

It is worth mentioning that the interview phase of the experiment involved only ten participants, selected without strict criteria to ensure demographic diversity or varied backgrounds. The sample consisted primarily of young adults between the ages of 18 and 30, with a gender distribution of eight males and two females. All participants were at least undergraduate students with ready access to information. Although their interest levels in the topics varied, most were at least somewhat familiar with the subjects discussed.

We acknowledge that this experiment lacks strong statistical rigor and does not aim to provide a representative evaluation of the embedding models. Rather, it highlights a well-known limitation of this type of evaluation: intruder detection tasks are inherently time- and resource-intensive. Despite these constraints, the results offer useful insights into the models' behavior and performance.

8.3 Automatic feature prediction

In every evaluated dataset, each item is described with at least one feature related to the domain of the problem, such as genres for movies and styles for music artists. For each item in the dataset, we predicted multiple features using the most recurrent ones of other k nearest items, with k ranging between $10, 15$, and 20. The selected features P_i were the ones presented on at least half of the neighbors.

For each prediction P_i, we checked if the selected features were correct, comparing them with the original subset of features of the item F_i, as shown in Equation 13. Using the average multiclass precision and recall (Equations 14 and 15, respectively), we computed the F1-score for each model, using the formula in Equation 16. Table 10 shows the results, with darker tone cells corresponding to the best results for each dataset. The best result for each dataset and k-value combination is highlighted in **bold**.

$$\text{TP} = \sum_{i}^{I}\sum_{p}^{P_i}\begin{cases}1, & \text{if } p \in F_i\\ 0, & \text{otherwise}\end{cases} \tag{13}$$

$$\text{Precision} = \frac{\text{TP}}{\sum_i^I(|P_i|)} \tag{14}$$

$$\text{Recall} = \frac{\text{TP}}{\sum_i^I(|F_i|)} \tag{15}$$

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{16}$$

When comparing the results, it is challenging to indicate a superior model, with the performance varying depending on the dataset. However, User2Vec performed best on BestBuy and for some values of k for the Delicious dataset, also being a close second on Last.FM. This indicates that the neural model discovered the most about the intrinsic content of all evaluated methods for these specific datasets. This outcome is interesting when we compare the results of the feature prediction task with those of the extrinsic experiment. In the latter, User2Vec was the worst representation model for every value of N and dataset. For the BestBuy dataset, for example, the NDCG when $N = 10$ was more than five times worse than that of BPR.

The opposite can be seen for the RecVAE model, which consistently outperformed every other model in the extrinsic task. For the automatic feature prediction, the neural network was, at most, the third-best model. In datasets such as Delicious, Last.FM and ML-25m, the model presented the worst F1-score for every value of k, with results being less than half the values obtained by the best model in some scenarios. This behavior, aligned with the good performance of User2Vec, highlights how the outcome for extrinsic and intrinsic tasks can drastically differ.

Dataset	k	Representation Model				
		ALS	BPR	I2V	U2V	RVA
Anime	**10**	0.472	**0.525**	0.493	0.477	0.480
	15	0.405	**0.464**	0.434	0.400	0.419
	20	0.400	**0.466**	0.442	0.400	0.421
BestBuy	**10**	0.084	0.121	0.301	**0.336**	0.083
	15	0.066	0.100	0.254	**0.285**	0.045
	20	0.066	0.101	0.251	**0.280**	0.036
Delicious	**10**	**0.136**	0.124	0.132	**0.136**	0.030
	15	0.095	0.084	0.089	**0.100**	0.009
	20	0.098	0.087	0.092	**0.104**	0.007
Last.FM	**10**	**0.465**	0.375	0.405	0.458	0.156
	15	**0.427**	0.327	0.342	0.425	0.089
	20	**0.437**	0.335	0.349	0.433	0.092
ML-25m	**10**	0.501	**0.516**	0.460	0.408	0.300
	15	0.457	**0.472**	0.405	0.344	0.218
	20	0.473	**0.488**	0.418	0.358	0.241

Table 10. F1-score in an automatic feature prediction task with different values of k.

In addition, we can see that those methods that reached better scores in Table 10 may differ from those of the intruder detection task. In the former, ALS was the most accurate model for the datasets Last.FM and Delicious when $k = 10$, while the latter was the least accurate for every dataset, including the aforementioned ones. This shows how even different intrinsic metrics can achieve varying results, especially when comparing subjective approaches to objective ones, since the former is more prone to human bias and personal interpretations.

8.4 Content-based ranking comparison

Lastly, we have calculated the three metrics for assessing the item embeddings' intrinsic quality using a content-based ranking comparison, as detailed in Section 5.2. Results for the Spearman correlation coefficient are shown in Table 11, for the Jaccard Index in Table 12, and NDCG in Table 13. For both the Jaccard Index and the NDCG, we used a neighborhood size k ranging from $\{10, 15, 20\}$.

Like the automatic feature prediction task, the performance varied widely according to the metric and dataset, with each model scoring higher in a specific case. For Spearman's ρ, Item2Vec was the best model for Anime and BestBuy datasets, contrary to what happened on the intruder detection and automatic feature prediction, in which the model was surpassed by BPR and User2Vec, depending on the dataset and task. User2Vec achieved the best results for Delicious and Last.FM, which is also impressive since its behavior on the intruder detection task for dataset Last.FM was poor, being the worst or second-worst model.

When limiting the observed neighborhood to a subset of items, as is performed on the Jaccard Index and NDCG, the scores were vastly different from Spearman's ρ. For the Jaccard Index, Item2Vec achieved the worst results for the Anime dataset and the best for Delicious and Last.FM. User2Vec presented the highest scores for BestBuy and competitive performance in the Anime dataset. Although ALS achieved some promising results for the Last.FM dataset, it was surpassed by every other model for every dataset. Finally, for the NDCG, the results were similar to the ones obtained in the automatic feature prediction (Table 10), which is expected given that both metrics limit the observed neighborhood and weigh their scores according to the content information.

The experiment shows how different metrics of ranking comparison can achieve different results for the same representation model and dataset. The use of a specific metric can vary according to the analysis's objective and the content-based representation's characteristics. When we want to evaluate the entire ranking, considering only the relative position of items, rank correlation metrics, such as Spearman's ρ, are more well-suited. In cases where only the quality of the neighborhood is important, disregarding the intensity of the item's similarity, metrics of set similarity are more recommended, such as the Jaccard Index. Lastly, when we are only interested in the neighborhood's items but want to weigh the results according to their similarity scores and rank positions, we can calculate utility-based metrics such as NDCG.

When comparing the content-based ranking metrics and the achieved results for the extrinsic evaluation, it is clear how representation models not useful for generating recommendations may still have value when used in intrinsic tasks. User2Vec, statistically proven as one of the worst methods for the top-N recommendation problem, presented excellent results for some datasets on the content-based analysis, especially when calculating the NDCG. Additionally, methods that achieve good outcomes in the extrinsic task may not hold the same performance on intrinsic tasks, as we can see with RecVAE. The SOTA model was the best algorithm in 90% of the datasets on the extrinsic evaluation, but presented poor results for every metric in the content-based ranking task, especially for datasets Delicious, Last.FM and ML-25m. The obtained results highlight how proper analysis of a model's intrinsic quality must be conducted independently from the extrinsic evaluation. Lastly, the differences in the obtained results with the intruder detection's accuracy also show how subjective approaches can lead to contrasting conclusions about the model's quality. This entire comparison demonstrates how a thoroughly performed analysis can lead to more knowledge about the behavior of representation models.

Dataset	**Representation Model**				
	ALS	**BPR**	**I2V**	**U2V**	**RVA**
Anime	0.144	0.250	**0.280**	0.186	0.209
BestBuy	0.473	0.465	**0.484**	0.480	0.481
Delicious	0.292	0.281	0.277	**0.293**	0.276
Last.FM	0.233	0.324	0.257	**0.355**	0.212
ML-25m	**0.257**	0.254	0.227	0.214	0.122

Table 11. Spearman's rank correlation coefficient ρ of the content-based ranking comparison.

Dataset	k	**Representation Model**				
		ALS	**BPR**	**I2V**	**U2V**	**RVA**
Anime	**10**	0.037	0.040	0.032	**0.042**	0.039
	15	0.035	**0.038**	0.031	**0.038**	0.036
	20	0.033	**0.037**	0.030	0.035	0.035
BestBuy	**10**	0.004	0.006	0.022	**0.026**	0.021
	15	0.005	0.007	0.027	**0.032**	0.021
	20	0.006	0.008	0.031	**0.037**	0.020
Delicious	**10**	0.002	0.002	**0.003**	**0.003**	0.001
	15	**0.003**	**0.003**	**0.003**	**0.003**	0.001
	20	**0.004**	**0.004**	**0.004**	**0.004**	0.002
Last.FM	**10**	0.022	0.018	**0.025**	0.019	0.006
	15	0.027	0.022	**0.030**	0.023	0.008
	20	0.031	0.026	**0.035**	0.027	0.009
ML-25m	**10**	**0.002**	**0.002**	0.001	0.001	0.001
	15	**0.002**	**0.002**	**0.002**	**0.002**	**0.002**
	20	0.002	**0.003**	0.002	0.002	0.002

Table 12. Jaccard Index@k of the content-based ranking comparison, with different values of k.

Regarding the content-based ranking evaluation technique, it is important to highlight that a content-based ranking comparison can lead to useful insights if the high-level

Dataset	k	Representation Model				
		ALS	BPR	I2V	U2V	RVA
Anime	**10**	0.469	**0.500**	0.468	0.459	0.470
	15	0.456	**0.488**	0.458	0.441	0.459
	20	0.448	**0.481**	0.451	0.429	0.452
BestBuy	**10**	0.115	0.149	0.321	**0.354**	0.177
	15	0.111	0.143	0.309	**0.341**	0.158
	20	0.108	0.140	0.300	**0.332**	0.147
Delicious	**10**	0.185	0.178	0.185	**0.186**	0.122
	15	0.185	0.178	0.184	**0.187**	0.120
	20	0.185	0.177	0.183	**0.187**	0.119
Last.FM	**10**	**0.436**	0.385	0.414	0.423	0.243
	15	**0.435**	0.385	0.412	0.423	0.243
	20	**0.435**	0.385	0.412	0.422	0.243
ML-25m	**10**	0.433	**0.444**	0.402	0.364	0.291
	15	0.426	**0.438**	0.395	0.356	0.287
	20	0.421	**0.433**	0.389	0.350	0.284

Table 13. NDCG@k of the content-based ranking comparison, with different values of k.

features of the items are good enough to represent their intrinsic value. Nevertheless, this may not be true for every scenario. For example, from Last.FM dataset, the top-3 closest artists using a content-based neighborhood for the same artists selected for the similarity tables would be the ones shown in Table 14.

As we can see, some neighboring items seem odd. It happens mainly because of two reasons: *(i)* common tags between items are not relevant to describe them, such as *metal* for *Elvis Presley* and *Madonna*, both artists that are known for other musical genres; and *(ii)* some tags are overused, and they do not help differentiate items, such as *rock* for *The Beatles*, or *pop* for *Eminem* and *Shakira*, and all their respective neighbors. Therefore, when using the ranking comparison technique, a good content-based representation is the key to achieving a valuable evaluation. Thus, the metrics can benefit from more robust methods to define an appropriate item similarity.

9 Conclusion

Embeddings with strong intrinsic meaning have the potential to benefit numerous tasks beyond recommendation. Since intrinsic evaluation has garnered attention in NLP, it is poised to become a focal point in recommender systems. However, intrinsic evaluations of item embeddings are often overlooked, as most studies prioritize extrinsic assessments. Even when intrinsic evaluations are performed, they often rely on human-centered metrics, which are not only time-consuming but also prone to biases and subjectivity.

The goal of this study was to address three key research questions. Initially, we wanted to define how intrinsic evaluation can be effectively conducted in the context of recommender systems (RQ1). For this, we introduced a guideline to assess the intrinsic quality of item embeddings. We began by explaining the use of similarity tables, a well-known evaluation method for recommenders, and adapting an NLP evaluation task for the context of recommender systems. Recognizing the limitations of subjective approaches that rely on human judgment, we also presented two objective evaluation metrics: a feature prediction task and a novel quantitative strategy based on content-based ranking comparison. For the latter, we provided various metrics to assess ranking quality. Using these evaluation approaches, we analyzed and compared the performance of five models that learn item embeddings using different techniques, e.g., matrix factorization and neural networks.

The next research question was to compare the obtained intrinsic results with those from extrinsic tasks (RQ2). We then conducted a traditional top-N ranking recommendation to assess the extrinsic quality of each model. The evaluation revealed a clear superiority of the RecVAE model, as well as similarities between the two matrix factorization methods and Item2Vec. RecVAE achieved the best results in 90% of the datasets, while the remaining algorithms excelled on specific datasets. Conversely, User2Vec performed poorly across all datasets in the extrinsic evaluation, emerging as the worst method.

Intriguingly, the intrinsic tasks were the opposite. User2Vec excelled in generating representations with intrinsic value, ranking first or second for most datasets for both subjective and objective approaches, and RecVAE was the worst model in many different intrinsic evaluation scenarios. Our findings highlight the necessity of careful intrinsic evaluation to avoid misleading impressions of a model's capabilities.

Finally, the third research question focused on understanding the differences between subjective and objective evaluation approaches (RQ3). Our findings reveal that while subjective tasks are often more intuitive, they present significant challenges in evaluating algorithms effectively. For instance, the generated similarity tables were not useful for properly comparing the model, as they exhibited similar behavior depending on the seed item. Even when the neighborhoods differed, evaluating them without bias was challenging. Similarly, for the intruder detection task, results varied considerably based on the dataset and the criteria used to construct the evaluated sets.

In contrast, objective evaluations offered several advantages. Metrics could be calculated across the entire dataset, providing more stable and reliable results while avoiding the biases inherent in subjective methods. This highlights the importance of incorporating objective approaches for a more comprehensive and unbiased assessment of embedding models.

For future research, we recommend conducting a detailed investigation into content-based representation and item-sorting methods to enhance the quality of the proposed strategies. Additionally, employing more domain-specific datasets with enriched feature sets and detailed descriptive attributes can further improve the robustness and depth of the results. We also plan to analyze why each algorithm performed better in each task. Finding which characteristics favor specific evaluation scenarios can help future researchers develop task-specific recommenders. Lastly, we suggest a similar study focusing on user embeddings, leveraging demographic information to address the common challenge of

Seed artist	**Most similar artists based on tags**		
	1st	**2nd**	**3rd**
The Beatles *classic rock, rock*	Muse *alt rock, rock*	Placebo *alt rock, alternative*	Duran Duran *80s, new wave*
Elvis Presley *classic rock, rock*	Madonna *dance, pop*	Paramore *female, rock*	Christina Aguilera *female, pop*
Eminem *rap, hip-hop*	Beyoncé *rnb, pop*	Hilary Duff *pop, dance*	Britney Spears *pop, female*
Linkin Park *nu metal, rock*	Death Cab for Cutie *indie, indie rock*	Red Hot Chilli Peppers *alt rock, rock*	Coldplay *alternative, rock*
Shakira *female, pop*	Nelly Furtado *female, pop*	Rihanna *pop, rnb*	Christina Aguilera *female, pop*

Table 14. Similarity table for Last.FM dataset using content-based representation.

data scarcity associated with users.

Ultimately, future research on item embeddings should prioritize evaluating their intrinsic quality. A thorough analysis can provide a deeper understanding of the models, offering valuable insights for tasks beyond recommendation and potentially accelerating the development of new models.

Declarations

Acknowledgements

The authors are grateful for the financial support provided by the Brazilian agencies FAPESP, CAPES and CNPq. Furthermore, we would also like to thank the anonymous reviewers of WebMedia'24 for their valuable feedback on our conference paper, which certainly improved the quality of the study.

Funding

This study was financed by the Brazilian Coordination for the Improvement of Higher Education Personnel (CAPES), Finance Code 88882.426978/2019-01; the Brazilian National Council for Scientific and Technological Development (CNPq); and the São Paulo Research Foundation (FAPESP), grant #2021/14591-7.

Authors' Contributions

PP: Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. TA: Conceptualization, Funding acquisition, Project administration, Supervision, Validation, Writing – original draft, Writing – review & editing.

Competing interests

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Availability of data and materials

All datasets used in this study are publicly available and can be found in the online repositories listed in Table 15. The source code is publicly available in the following repository: `https://github.com/UFSCar-LaSID/recsys-intrinsic-evaluation`.

References

Barkan, O. and Koenigstein, N. (2016). Item2Vec: Neural item embedding for collaborative filtering. In *IEEE 26th International Workshop on Machine Learning for Signal Processing*, MLSP 2016, pages 1–6, Vietri sul Mare, Italy. IEEE. DOI: 10.1109/MLSP.2016.7738886.

Baroni, M., Dinu, G., and Kruszewski, G. (2014). Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, ACL '14, page 238–247, Baltimore, MD, USA. Association for Computational Linguistics. DOI: 10.3115/v1/P14-1023.

Bengio, Y., Ducharme, R., Vincent, P., and Janvin, C. (2003). A neural probabilistic language model. *The Journal of Machine Learning Research*, 3:1137–1155. DOI: 10.5555/944919.944966.

Bennett, J. and Lanning, S. (2007). The netflix prize. In *Proceedings of KDD Cup and Workshop*, KDD '07. Association for Computing Machinery. DOI: 10.1145/1327942.1327945.

Bobadilla, J., Ortega, F., Hernando, A., and Gutiérrez, A. (2013). Recommender systems survey. *Knowledge-Based Systems*, 46:109–132. DOI: 10.1016/j.knosys.2013.03.012.

Cantador, I., Brusilovsky, P., and Kuflik, T. (2011). 2nd workshop on information heterogeneity and fusion in recommender systems (hetrec 2011). In *Proceedings of the 5th ACM conference on Recommender systems*, RecSys 2011, New York, NY, USA. ACM.

Caselles-Duprés, H., Lesaint, F., and Royo-Letelier, J. (2018). Word2vec applied to recommendation: hyperparameters matter. In *Proceedings of the 12th ACM Conference on Recommender Systems*, RecSys '18, pages 352–356, Vancouver, Canada. Association for Computing Machinery. DOI: 10.1145/3240323.3240377.

Chang, C., Zhou, J., Weng, Y., Zeng, X., Wu, Z., Wang, C.-D., and Tang, Y. (2023). KGTN: Knowledge graph transformer network for explainable multi-category item rec-

Dataset	Authors	Source
Anime	CooperUnion [2017]	`kaggle.com/datasets/CooperUnion/anime-recommendations-database`
BestBuy	Glider *et al.* [2012]	`kaggle.com/competitions/acm-sf-chapter-hackathon-big`
BookCross	Ziegler *et al.* [2005]	`kaggle.com/competitions/acm-sf-chapter-hackathon-big`
CiaoDVD	Guo *et al.* [2014]	`guoguibing.github.io/librec/datasets.html`
Delicious	Cantador *et al.* [2011]	`grouplens.org/datasets/hetrec-2011/`
FilmTrust	Guo *et al.* [2013]	`guoguibing.github.io/librec/datasets.html`
Last.FM	Cantador *et al.* [2011]	`grouplens.org/datasets/hetrec-2011/`
ML-25m	Harper and Konstan [2015]	`grouplens.org/datasets/movielens/`
NetflixPrize	Bennett and Lanning [2007]	`kaggle.com/datasets/netflix-inc/netflix-prize-data`
RRocket	Zykov *et al.* [2022]	`kaggle.com/datasets/retailrocket/ecommerce-dataset`

Table 15. Source of the datasets analyzed in this study.

ommendation. *Knowledge-Based Systems*, 278:110854. DOI: 10.1016/j.knosys.2023.110854.

Chen, H., Wang, Z., Huang, F., Huang, X., Xu, Y., Lin, Y., He, P., and Li, Z. (2022). Generative adversarial framework for cold-start item recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, page 2565–2571, Anchorage, AK, USA. Association for Computing Machinery. DOI: 10.1145/3477495.3531897.

CooperUnion (2017). Anime recommendations database. Available at:`https://www.kaggle.com/datasets/CooperUnion/anime-recommendations-database`.

de Souza P. Moreira, G., Jannach, D., and da Cunha, A. M. (2019). On the importance of news content representation in hybrid neural session-based recommender systems. *IEEE Access*, 7:169185–169203. DOI: 10.1109/ACCESS.2019.2954957.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:1–30. DOI: 10.5555/1248547.1248548.

Ding, C., Zhao, Z., Li, C., Yu, Y., and Zeng, Q. (2023). Session-based recommendation with hypergraph convolutional networks and sequential information embeddings. *Expert Systems with Applications*, 223(119875):1–11. DOI: 10.1016/j.eswa.2023.119875.

Eck, D., Lamere, P., Bertin-Mahieux, T., and Green, S. (2007). Automatic generation of social tags for music recommendation. In *Proceedings of the 20th International Conference on Neural Information Processing Systems*, NIPS 2007, pages 385–392, Vancouver, Canada. Curran Associates Inc.. DOI: 10.5555/2981562.2981611.

Faruqui, M., Tsvetkov, Y., Rastogi, P., and Dyer, C. (2016). Problems with evaluation of word embeddings using word similarity tasks. In *Proceedings of the 1st Workshop on Evaluating Vector Space Representations for NLP*, pages 30–35, Berlin, Germany. Association for Computational Linguistics. DOI: 10.18653/v1/W16-2506.

Filho, R. J. R., Wehrmann, J., and Barros, R. C. (2017). Leveraging deep visual features for content-based movie recommender systems. In *Proceedings of the 2017 International Joint Conference on Neural Networks*, IJCNN 2017, pages 604–611, Anchorage, AK, USA. IEEE. DOI: 10.1109/IJCNN.2017.7965908.

Firan, C. S., Nejdl, W., and Paiu, R. (2007). The benefit of using tag-based profiles. In *Proceedings of the 5th Latin American Web Conference*, LA-WEB '07, pages 32–41, Santiago, Chile. IEEE Computer Society. DOI: 10.1109/LA-WEB.2007.24.

FU, P., hua LV, J., long MA, S., and jie LI, B. (2017). Attr2vec: a neural network based item embedding method. In *Proceedings of the 2nd International Conference on Computer, Mechatronics and Electronic Engineering*, CMEE 2017, pages 300–307, Xiamen, China. DEStech Publications. DOI: 10.12783/dtcse/cmee2017/19993.

Gao, Y., Sheng, T., Xiang, Y., Xiong, Y., Wang, H., and Zhang, J. (2023). Chat-REC: Towards interactive and explainable LLMs-augmented recommender system. *arXiv:*, 2303.14524:1–17. DOI: 10.48550/arXiv.2303.14524.

Gladkova, A. and Drozd, A. (2016). Intrinsic evaluations of word embeddings: What can we do better? In *Proceedings of the 1st Workshop on Evaluating Vector Space Representations for NLP*, pages 36–42, Berlin, Germany. Association for Computational Linguistics. DOI: 10.18653/v1/W16-2507.

Glider, G. M., Beladia, N., and Kolegraff, N. (2012). Data mining hackathon on big data (7gb) best buy mobile web site. Available at:`https://www.kaggle.com/competitions/acm-sf-chapter-hackathon-big/overview`.

Grbovic, M., Radosavljevic, V., Djuric, N., Bhamidipati, N., Savla, J., Bhagwan, V., and Sharp, D. (2015). E-commerce in your inbox: Product recommendations at scale. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, pages 1809–1818, Sydney, Australia. Association for Computing Machinery. DOI: 10.1145/2783258.2788627.

Greenstein-Messica, A., Rokach, L., and Friedman, M. (2017). Session-based recommendations using item embedding. In *Proceedings of the 22nd International Conference on Intelligent User Interfaces*, IUI '17, pages 629–633, Limassol, Cyprus. Association for Computing Machinery. DOI: 10.1145/3025171.3025197.

Guo, G., Zhang, J., Thalmann, D., and Yorke-Smith, N. (2014). ETAF: An extended trust antecedents framework for trust prediction. In *Proceedings of the 2014 International Conference on Advances in Social Networks Analysis and Mining*, ASONAM, pages 540–547, Beijing, China. IEEE. DOI: 10.1109/ASONAM.2014.6921639.

Guo, G., Zhang, J., and Yorke-Smith, N. (2013). A novel bayesian similarity measure for recommender systems. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, IJCAI '13, page 2619–

2625. AAAI Press. DOI: 10.5555/2540128.2540506.

Harper, F. M. and Konstan, J. A. (2015). The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 5(4). DOI: 10.1145/2827872.

Hasanzadeh, S., Fakhrahmad, S. M., and Taheri, M. (2020). Review-based recommender systems: A proposed rating prediction scheme using word embedding representation of reviews. *The Computer Journal*, bxaa044(;):1–10. DOI: 10.1093/comjnl/bxaa044.

Hernando, A., Bobadilla, J., and Ortega, F. (2016). A non negative matrix factorization for collaborative filtering recommender systems based on a bayesian probabilistic model. *Knowledge-Based Systems*, 97(C):188—202. DOI: 10.1016/j.knosys.2015.12.018.

Hidasi, B., Karatzoglou, A., Baltrunas, L., and Tikk, D. (2016). Session-based recommendations with recurrent neural networks. In *Proceedings of the International Conference on Learning Representations*, ICLR 2016, pages 1–10, San Juan, Puerto Rico. OpenReview. DOI: 10.48550/arXiv.1511.06939.

Hu, Y., Koren, Y., and Volinsky, C. (2008). Collaborative filtering for implicit feedback datasets. In *Proceedings of the 8th IEEE International Conference on Data Mining*, ICDM '08, pages 263–272, Pisa, Italy. IEEE Computer Society. DOI: 10.1109/ICDM.2008.22.

Júnior, S. M. and Manzato, M. G. (2015). Collaborative filtering based on semantic distance among items. In *Proceedings of the 21st Brazilian Symposium on Multimedia and the Web*, WebMedia '15, page 53–56, Manaus, Brazil. Association for Computing Machinery. DOI: 10.1145/2820426.2820466.

Khsuro, S., Ali, Z., and Ullah, I. (2016). Recommender systems: Issues, challenges, and research opportunities. In *Proceedings of the 7th International Conference on Information Science and Applications*, ICISA 2016, pages 1179–1189, Ho Chi Minh, Vietnam. Springer Science+Business Media. DOI: 10.1007/978-981-10-0557-2_112.

Kim, J., Kim, J., Yeo, K., Kim, E., On, K.-W., Mun, J., and Lee, J. (2024). General item representation learning for cold-start content recommendations. *arXiv:*, 2404.13808:1–14. DOI: 10.48550/arXiv.2404.13808.

Koren, Y., Bell, R., and Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37. DOI: 10.1109/MC.2009.263.

Le, Q. and Mikolov, T. (2014). Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning*, ICML 2014, pages 1188–1196, Beijing, China. JMLR.org. DOI: 10.5555/3044805.3045025.

Lisena, P., Meroño-Peñuela, A., and Troncy, R. (2022). MIDI2vec: Learning MIDI embeddings for reliable prediction of symbolic music metadata. *Semantic Web*, 13(3):357–377. DOI: 10.3233/SW-210446.

Liu, J., Liu, C., Zhou, P., Ye, Q., Chong, D., Zhou, K., Xie, Y., Cao, Y., Wang, S., You, C., and S.Yu, P. (2023). LLMRec: Benchmarking large language models on recommendation task. *arXiv:*, 2308.12241:1–13. DOI: 10.48550/arXiv.2308.12241.

Lu, J., Wu, D., Mao, M., Wang, W., and Zhang, G. (2015). Recommender system application developments: A survey. *Decision Support Systems*, 74:12–32. DOI: 10.1016/j.dss.2015.03.008.

Mikolov, T., Sutskever, I., Chen, K., Conrado, G., and Dan, J. (2013). Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems*, NIPS 2013, pages 3111–3119, Stateline, NV, USA. Curran Associates Inc.. DOI: 10.5555/2999792.2999959.

Musto, C., Lops, P., de Gemmis, M., and Semeraro, G. (2017). Semantics-aware recommender systems exploiting linked open data and graph-based features. *Knowledge-Based Systems*, 136:1–14. DOI: 10.1016/j.knosys.2017.08.015.

Ozsoy, M. G. (2016). From word embeddings to item recommendation. *arXiv:*, 1601.01356:1–8. DOI: 10.48550/arXiv.1601.01356.

Pazzani, M. J. and Billsus, D. (2007). Content-based recommendation systems. *The Adaptive Web*, Lecture Notes in Computer Science, vol 4321:325–341. DOI: 10.1007/978-3-540-72079-9_10.

Pires, P. R., Rizzi, B. B., and Almeida, T. A. (2024). Why ignore content? a guideline for intrinsic evaluation of item embeddings for collaborative filtering. In *Proceedings of the 30th Brazilian Symposium on Multimedia and the Web*, WebMedia 2024, pages 345–354, Juiz de Fora, Brazil. Sociedade Brasileira de Computação. DOI: 10.5753/webmedia.2024.243199.

Qiu, Y., Li, H., Li, S., Jiang, Y., Hu, R., and Yang, L. (2018). Revisiting correlations between intrinsic and extrinsic evaluations of word embeddings. In *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data*, CCL 2018, pages 209–221, Changsha, China. Springer. DOI: 10.1007/978-3-030-01716-3_18.

Řehůřek, R. and Sojka, P. (2010). Software framework for topic modelling with large corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, LREC 2010, pages 45–50, Valletta, Malta. European Language Resources Association (ELRA). DOI: 10.13140/2.1.2393.1847.

Rendle, S., Freudenthaler, C., Gantner, Z., and Schmidt-Thieme, L. (2009). BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, UAI '09, pages 452–461, Montreal, Canada. AUAI Press. DOI: 10.5555/1795114.1795167.

Rendle, S., Krichene, W., Zhang, L., and Anderson, J. (2020). Neural collaborative filtering vs. matrix factorization revisited. In *Proceedings of the 14th ACM Conference on Recommender Systems*, RecSys '20, pages 240–248, Virtual Event, Brazil. Association for Computing Machinery. DOI: 10.1145/3383313.3412488.

Rendle, S., Krichene, W., Zhang, L., and Koren, Y. (2022). Revisiting the performance of ials on item recommendation benchmarks. In *Proceedings of the 16th ACM Confer-*

ence on Recommender Systems, RecSys '22, pages 427–435, Seattle, WA, USA. Association for Computing Machinery. DOI: 10.1145/3523227.3548486.

Sarwar, B. M., Karypis, G., Konstan, J. A., and Riedl, J. T. (2000). Application of dimensionality reduction in recommender system - a case study. In *Proceedings of the 9th WebKDD Workshop on Web Mining for e-commerce*, WebKDD '00, pages 1–12, Boston, Massachusetts, USA. Association for Computing Machinery. DOI: 10.21236/ada439541.

Schnabel, T., Labutov, I., Mimno, D., and Joachims, T. (2015). Evaluation methods for unsupervised word embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, EMNLP 2015, pages 298–307, Lisbon, Portugal. Association for Computational Linguistics. DOI: 10.18653/v1/D15-1036.

Senel, L. K., Utlu, I., Yücesoy, V., Koç, A., and Çukur, T. (2018). Semantic structure and interpretability of word embeddings. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(10):1769–1779. DOI: 10.1109/TASLP.2018.2837384.

Shani, G. and Gunawardana, A. (2011). Evaluating recommendation systems. In Ricci, F., Rokach, L., Shapira, B., and Kantor, P. B., editors, *Recommender Systems Handbook*, chapter 8, pages 257–259. Springer US, New York, NY, USA. DOI: 10.1007/978-0-387-85820-3.

Shenbin, I., Alekseev, A., Tutubalina, E., Malykh, V., and Nikolenko, S. I. (2020). RecVAE: A new variational autoencoder for top-n recommendations with implicit feedback. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, WSDM '20, page 528–536, New York, NY, USA. Association for Computing Machinery. DOI: 10.1145/3336191.3371831.

Sidana, S., Trofimov, M., Horodnytskyi, O., Laclau, C., Maximov, Y., and Amini, M.-R. (2021). User preference and embedding learning with implicit feedback for recommender systems. *Data Mining and Knowledge Discovery*, 35:568–592. DOI: 10.1007/s10618-020-00730-8.

Siswanto, A. V., Tjong, L., and Saputra, Y. (2018). Simple vector representations of e-commerce products. In *2018 International Conference on Asian Language Processing*, IALP 2018, pages 368–372, Bandung, Indonesia. IEEE. DOI: 10.1109/IALP.2018.8629245.

Song, Y., Zhang, L., and Giles, C. L. (2011). Automatic tag recommendation algorithms for social recommender systems. *ACM Transactions on the Web*, 4(1):4:1–4:31. DOI: 10.1145/1921591.1921595.

Tang, J. and Wang, K. (2018). Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the 11th ACM International Conference on Web Search and Data Mining*, WSDM '18, pages 565–573, Marina Del Rey, CA, USA. Association for Computing Machinery. DOI: 10.1145/2939672.2939673.

Vasile, F., Smirnova, E., and Conneau, A. (2016). Meta-prod2vec: Product embeddings using side-information for recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16, pages 225–232, Boston, Massachusetts, USA. Association for Computing Machinery. DOI: 10.1145/2959100.2959160.

Wang, D., Xu, G., and Deng, S. (2017). Music recommendation via heterogeneous information graph embedding. In *Proceedings of the 2017 International Joint Conference on Neural Networks*, IJCNN 2017, pages 596–603, Anchorage, AK, USA. IEEE. DOI: 10.1109/IJCNN.2017.7965907.

Wang, J. and Lv, J. (2020). Tag-informed collaborative topic modeling for cross domain recommendations. *Knowledge-Based Systems*, 203:106119. DOI: 10.1016/j.knosys.2020.106119.

Wang, Q., Yin, H., Wang, H., Nguyen, Q. V. H., Huang, Z., and Cui, L. (2019). Enhancing collaborative filtering with generative augmentation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '19, page 548–556, Anchorage, AK, USA. Association for Computing Machinery. DOI: 10.1145/3292500.3330873.

Wang, T., Brovman, Y. M., and Madhvanath, S. (2021). Personalized embedding-based e-commerce recommendations at ebay. *arXiv:*, 2102.06156:1–9. DOI: 10.48550/arXiv.2102.06156.

Werneck, H., Silva, N., Viana, M. C., ao, F. M., Pereira, A. C. M., and Rocha, L. (2020). A survey on point-of-interest recommendation in location-based social networks. In *Proceedings of the Brazilian Symposium on Multimedia and the Web*, WebMedia '20, page 185–192, São Luís, Brazil. Association for Computing Machinery. DOI: 10.1145/3428658.3430970.

Yang, D., Li, N., Zou, L., and Ma, H. (2022). Lexical semantics enhanced neural word embeddings. *Knowledge-Based Systems*, 252:109298. DOI: 10.1016/j.knosys.2022.109298.

Yu, J., Yin, H., Xia, X., Chen, T., Li, J., and Huang, Z. (2024). Self-supervised learning for recommender systems: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 36:335–355. DOI: 10.1109/TKDE.2023.3282907.

Zarzour, H., Al-Sharif, Z. A., and Jararweh, Y. (2019). RecDNNing: a recommender system using deep neural network with user and item embeddings. In *Proceedings of the 10th International Conference on Information and Communication Systems*, ICICS 2019, pages 99–103, Irbid, Jordan. IEEE. DOI: 10.1109/IACS.2019.8809156.

Zhang, F., Yuan, N. J., Lian, D., Xie, X., and Ma, W.-Y. (2016). Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 353–362, San Francisco, CA, USA. Association for Computing Machinery. DOI: 10.1145/2939672.2939673.

Zhang, S., Yao, L., Sun, A., and Tay, Y. (2019). Deep learning based recommender system: A survey and new perspectives. *ACM Comput. Surv.*, 52(1):5:1–5:35. DOI: 10.1145/3285029.

Zhao, X., Wang, M., Zhao, X., Li, J., Zhou, S., Yin, D., Li, Q., Tang, J., and Guo, R. (2023). Embedding in recommender systems: A survey. *arXiv:*, 2310.18608:1–42. DOI: 10.48550/arXiv.2310.18608.

Ziegler, C.-N., McNee, S. M., Konstan, J. A., and Lausen,

G. (2005). Improving recommendation lists through topic diversification. In *Proceedings of the 14th International Conference on World Wide Web*, WWW '05, page 22–32, New York, NY, USA. Association for Computing Machinery. DOI: 10.1145/1060745.1060754.

Zykov, R., Artem, N., and Alexander, A. (2022). Retailrocket recommender system dataset. DOI: 10.34740/KAGGLE/DSV/4471234.