Single-Branch Network Architectures to Close the Modality Gap in Multimodal Recommendation

CHRISTIAN GANHÖR, Institute of Computational Perception, Johannes Kepler University Linz, Austria

MARTA MOSCATI, Institute of Computational Perception, Johannes Kepler University Linz, Austria

ANNA HAUSBERGER, Institute of Computational Perception, Johannes Kepler University Linz, Austria

SHAH NAWAZ, Institute of Computational Perception, Johannes Kepler University Linz, Austria

MARKUS SCHEDL*, Institute of Computational Perception, Johannes Kepler University Linz and AI Lab, Linz Institute of Technology, Austria

Traditional recommender systems rely on collaborative filtering (CF), using past user–item interactions to help users discover new items in a vast collection. In *cold start*, i. e., when interaction histories of users or items are not available, content-based recommender systems (CBRSs) use side information instead. Most commonly, user demographics and item descriptions are used for user and item cold start, respectively. Hybrid recommender systems (HRSs) often employ multimodal learning to combine collaborative and user and item side information, which we jointly refer to as *modalities*. Though HRSs can provide recommendations when some modalities are missing, their quality degrades. In this work, we utilize *single-branch neural networks* equipped with *weight sharing*, *modality sampling*, and *contrastive loss* to provide accurate recommendations even in missing modality scenarios, including cold start. Compared to multi-branch architectures, the weights of the encoding modules are shared for all modalities; in other words, all modalities are encoded using the same neural network. This, together with the contrastive loss, is essential in reducing the modality gap, while the modality sampling is essential in modeling missing modality during training. Simultaneously leveraging these techniques results in more accurate recommendations. We compare these networks with multi-branch alternatives and conduct extensive experiments on the MovieLens 1M, Music4All-Onion, and Amazon Video Games datasets. Six accuracy-based and four beyond-accuracy-based metrics help assess the recommendation quality for the different training paradigms and their hyperparameters on single- and multi-branch networks in warm-start and missing modality scenarios. We quantitatively and qualitatively study the effects of these different aspects on bridging the modality gap. Our results show that single-branch networks provide competitive recommendation quality in warm start, and significantly better performance in missing modality scenarios. Moreover, our study of modality sampling and contrastive loss on both single- and multi-branch architectures indicates a consistent positive impact on accuracy metrics across all datasets. Overall, the three training paradigms collectively encourage modalities of the same item to be embedded closer together than those of different items, as measured by Euclidean distance and cosine similarity. This results in embeddings that are less distinguishable and more interchangeable, as indicated by a 7-20% drop in modality prediction accuracy. Our full experimental setup, including

---

*Corresponding author.

Authors' Contact Information: Christian Ganhör, christian.ganhoer@jku.at, Institute of Computational Perception, Johannes Kepler University Linz, Linz, Austria; Marta Moscati, marta.moscati@jku.at, Institute of Computational Perception, Johannes Kepler University Linz, Linz, Austria; Anna Hausberger, anna.hausberger@jku.at, Institute of Computational Perception, Johannes Kepler University Linz, Linz, Austria; Shah Nawaz, shah.nawaz@jku.at, Institute of Computational Perception, Johannes Kepler University Linz, Linz, Austria; Markus Schedl, markus.schedl@jku.at, Institute of Computational Perception, Johannes Kepler University Linz and AI Lab, Linz Institute of Technology, Linz, Austria.

---

training and evaluating code for all algorithms, their hyperparameter configurations, and our result analysis notebooks, is available at https://github.com/hcai-mms/single-branch-networks.

1 Introduction

Recommender systems (RSs) offer an important tool for users to discover interesting items in the vast and ever-increasing collections of, e. g., movies, songs, and shopping items, alleviating users' information overload. Although many RSs still build on collaborative filtering (CF), using past user–item interactions, scientific developments related to multimedia content analysis enabled content-based recommenders (CBRSs). These RSs leverage a broad set of media types (text, audio, image, video) and are capable of effectively dealing with *cold-start* scenarios [12], i. e., when user and/or item interaction histories are unavailable. While CF-based RSs struggle due to their dependence on interaction histories in cold start, the performance of CBRSs might be limited by not using collaborative information in warm start, i. e., when interactions are available in sufficient quantities. Thus, hybrid RSs (HRSs) overcome the shortcomings of both CFs and CBRSs, by integrating collaborative signals and additional side information of users (e. g., age, gender, personal descriptions) and/or items (e. g., audio for music RSs; video for movie RSs) to generate recommendations in warm- and cold-start scenarios. For this purpose, state-of-the-art HRSs commonly utilize deep neural networks (DNNs) along with multimodal learning approaches. These networks often contain modality-specific subnetworks (*branches*) for users and items alike, which encode and transform available modalities. Finally, the resulting modality representations per entity (users or items) are aggregated and used in downstream recommendation tasks [11, 25, 54, 59, 72]. Based on their architecture, we refer to this type of HRS as multi-branch recommenders, and outline a common structure of them in Figure 2. When subsets of the modalities used in training such models are unavailable at inference, the quality of recommendations often declines. This issue is referred to as *missing modality problem*. To address it, Ganhör et al. [17] propose a multimodal **Si**ngle-**Bra**nch embedding network for **R**ecommendation (SiBraR, pronounced "zebra"). As outlined in Figure 1, it leverages only a single shared branch per entity which maps all the modalities of the same entity to a similar region in the shared embedding space. In addition to weight sharing across modalities, SiBraR employs modality sampling and a contrastive loss between multiple modalities to further realize a shared embedding space. This design reduces the dependence on the availability of all modalities.

While single-branch architectures show promising results in warm- and cold- start scenarios as well as in scenarios of missing modalities, there is a need for further investigating how they work, and to compare them with multi-branch architectures. Moreover, their three main components: weight sharing, modality sampling, and contrastive loss, need to receive additional attention to study which of them, individually and jointly, contribute most to a strong recommendation performance.

In this work, we aim to develop an in-depth understanding of single- and multi-branch architectures for multimodal representation learning for recommendation. Thus, we task ourselves with answering the following research questions:

RQ1. How does SiBraR compare against multi-branch architectures in terms of accuracy and beyond-accuracy performance, in warm-start, missing modality, and cold-start scenarios?

RQ2. How do different training paradigms employed in SiBraR effect its performance?

RQ3. How can we characterize the modality gap for SiBraR and MuBraR?

To achieve our objectives, we introduce multimodal **Mu**lti-**Bra**nch embedding network for **R**ecommendation (MuBraR), serving as a common baseline for multi-branch recommender systems. Moreover, we propose two further variants, $MuBraR_S$ and $MuBraR_{SC}$, employing modality sampling without and with a contrastive loss, respectively, as used in SiBraR. Doing so, we discover the effects of the individual components on performance and the embedding space. We apply the models on three different recommendation domains (music, movie, and e-commerce) mimicking warm-start as well as user and item cold-start tasks. We evaluate them on six accuracy metrics and four beyond-accuracy metrics. The accuracy is measured with Precision, Recall, F-score, Normalized Discounted Cumulative Gain (NDCG), Mean Average Precision (MAP), and Mean Reciprocal Rank (MRR). The beyond-accuracy metrics consist of Coverage, Average Popularity of Long Tail Items (APLT), Average Recommendation Popularity (ARP), Popularity Lift (PL) and are used to describe induced model biases. Moreover, we thoroughly evaluate model performances in case of missing modalities. As SiBraR's main intention is to effectively reduce the modality gap, we qualitatively and quantitatively investigate the embedding spaces of single- and multi-branch paradigms. Finally, we take a deep dive into the contrastive loss and analyze how its parameters affect the recommendation quality as well as the embedding space.

We summarize the key contributions of our work as follows:

- We quantitatively compare single- to multi-branch approaches in extensive experiments on three datasets, in warm-start and missing-modality scenarios, including cold start.
- We shed light on the factors contributing to strong recommendation performance in single- and multi-branch models.
- We qualitatively and quantitatively investigate the embedding spaces of single- and multi-branch approaches, especially with respect to modality gap.
- We characterize the importance of the contrastive loss with a detailed analysis.

Our analyses show that single-branch recommenders have a clear advantage over multi-branch recommenders in cold start and missing modality, and that for both architectures the use of contrastive loss is paramount for achieving strong results. In our experiments single- and multi-branch architectures proved out to require extensive hyperparameter optimization. Therefore our detailed analysis of the impact of the hyperparameters of the contrastive loss can help practitioners in identifying suitable hyperparameter ranges. In terms of modality gap of embedding spaces, we show that both modality sampling and contrastive loss force modality embeddings of the same items to be closer to each other than to embeddings of other items.

The study of single-branch networks for recommendation opens many interesting directions of research. As such, it allows for studying missing modality during training, which the network inherently supports. Further, the impact of different strategies to fuse modalities, e. g., by attention, may be of interest in future works.

The remaining part of this paper is structured as follows: In Section 2, we discuss related works on cold-start recommendation and multimodal learning. We outline SiBraR and MuBraR in Section 3, followed by descriptions of the experimental setup in Section 4. All our experiments' quantitative and qualitative results are presented and discussed in Section 5. Finally, we conclude by considering practical implications of our work, as well as reviewing

limitations in Section 6. We also release our full experimental setup, including the training and evaluating pipelines for all algorithms, the hyperparameter configurations, and our result analysis code, at https://github.com/hcai-mms/single-branch-networks.

2   Related Work

In this section, we study relevant previous work, which falls within two strands of research: HRSs for cold start and multimodal representation learning.

2.1   Content-Based Cold-Start Recommendation

One of the main challenges of RSs [61, 66] is providing relevant recommendations to users with few or no past interactions, or to effectively recommend items that have been interacted with by few or no users; these scenarios are referred to user and item cold start, respectively. We refer the reader to Panda and Ray [49] for a recent literature review on cold-start recommendation.

Although there is no unanimous consensus on the number of interactions required to consider a user or an item as "cold", in this work, we focus on the *strict* cold-start scenario, i. e., when users or items have no interactions. To mitigate the low accuracy of recommendations in this scenario, RSs typically complement the user–item interaction data with side information, such as information on the user demographics, or representations of the content of the items. Several techniques have been proposed to leverage the additional information. They can be categorized into explicit and implicit methods [76].

Explicit techniques. Methods in this category combine the recommendation loss with additional loss terms. Specifically, several works proposed to additionally minimize the distance of latent representations of side information to latent representations of the interaction profiles. Magron and Févotte [37] and Cao et al. [10] explicitly model the distance by using the Euclidean distance between the latent representations as a loss term. While the former use fully-connected NNs to compute the latent representations, the latter leverage graph neural networks (GNNs). In a similar way, Barkan et al. [5] and Zhu et al. [84] include the Mean Square Error (MSE) between the latent representations as an additional loss term to reduce the distance between them. Another explicit way to reduce the distance between the latent representations is by using contrastive learning. Wei et al. [77] propose the use of a contrastive loss term on the latent embeddings resulting from the deep NNs that encode side information and interaction profiles. Li et al. [30] propose a GNN-based RSs with a contrastive loss term that uses as positive pairs the items that often co-occur in the interaction profiles of the same users. Other methods to learn latent representations of side information that are effective for recommendation include reconstruction losses. For instance, Wu et al. [79] propose the use of two autoencoders, one for item content and one for user interaction data, trained to minimize the combination of a recommendation loss term and reconstruction loss terms. Further works explore the use of categorical side information to learn similar latent representations within a category. For instance, in the context of music recommendation, Pulis and Bajada [53] maximize the similarity of audio representations of music tracks of the same music genre. In contrast, in the context of sequential recommendation, Shalaby et al. [67] complement the recommendation loss of a transformer-based sequential RS with a loss term for predicting item category. Moreover, within the category of explicit methods, Wang et al. [76] assume that the content of items produced at different times evolves continuously over time; they, therefore, introduce a loss term to align the representations of interpolations of items with the interpolation of the item representations. This technique aims to provide effective representations for items whose content is out of distribution compared to the training data, specifically

for cold-start items. The advantage of explicit methods is that the required similarity between representations of the interactions and the side information is used directly to optimize the model. This, however, comes at the cost of tuning additional hyperparameters, such as the weight of the similarity loss relative to the recommendation loss.

Implicit techniques. In the category of implicit methods, RSs leverage side information for cold start by relying on training strategies instead of explicit terms in the loss function. The RSs proposed by R et al. [54] and by Volkovs et al. [72] rely on multi-branch architectures that encode interaction information and one or more side information with separate NNs; the resulting representations are concatenated to obtain the representation used to provide recommendations. The difference between those two works is that Volkovs et al. [72] further apply dropout to mimic cold-start during training. Other architectures, such as those proposed by Cai et al. [9] and by Behar et al. [6], make use of heterogeneous graphs to simultaneously encode information on the interactions and the similarity between side information of different users or items. In a way similar to autoencoder approaches, Magron and Févotte [37] propose a NN that takes side information as input and aims at reconstructing the collaborative embedding, while Raziperchikolaei et al. [56] propose a NN that takes content as input and outputs the weights of the first layer of the encoder of the collaborative embedding. Finally, as large language models (LLMs) have demonstrated their ability in zero-shot learning, Gong et al. [18] and Sanner et al. [65] proposed the use of pre-trained LLMs to obtain item representations to be leveraged in cold-start scenarios. Compared to explicit methods, implicit methods have the advantage that representations are optimized directly for the recommendation task, without additional loss terms for the similarity. Due to the sparsity and noise in the interaction data, however, this signal might be too weak to enforce the required similarity between representations of the interactions and the side information.

2.2 Multimodal Learning

Multimodal learning leverages information from multiple modalities, such as audio, image, or text, to improve the performance on various machine learning (ML) tasks, such as classification, ranking, or verification [4, 33, 80]. Although multimodal tasks differ, the way they are usually addressed is very similar and relies on learning joint representations that either combine, e. g., concatenate, multiple modalities, or represent modalities separately, but in a way such that they are close in the embedding space if corresponding to the same entity. Several multimodal methods explored the use of NNs to map multimodal information to joint representations, e.g., multi-branch NNs use separate independent and modality-specific NNs to map each modality to a joint embedding space [3, 45, 46, 57, 62]. Recently, multi-branch architectures have also been extended with the use of transformers [34, 68, 71, 80]. Such multi-branch methods have achieved remarkable performance using modality-complete data, i. e., when all modalities are available for training and evaluation. However, recent studies have found that such methods lack inherent robustness to missing modalities, leading to a significant drop in performance when certain modalities are missing for the classification task [35]. Since recommendation systems often deal with incomplete data (e. g., missing user preferences), multi-branch Transformer architectures may not be the most effective choice for such applications. Given the significance of multimodal learning, recent years have seen an increased interest in studies addressing missing modalities for various deep learning tasks [28, 29, 32, 36, 64, 75, 83]. This growing interest has also broadened to multimodal RSs [38, 50, 74], where handling missing modalities is critical to improve performance and robustness. Recently, single-branch models [63, 69], i. e., models that share the same network across multiple modalities, have shown promising results in multimodal learning. Although effective for other multimodal learning tasks, applying single-branch architectures to the domain of recommendation poses the additional challenge of combining dense and sparse data, such as images and user-item interactions. Our work addresses this challenge by

adopting SiBraR, a novel multimodal RS based on a single-branch architecture [17], and analyzing the effectiveness of single-branch architectures in missing modality scenarios, which are related to cold start in recommendation.

3 Methodology

This section introduces multimodal **Si**ngle-**Bra**nch embedding network for **R**ecommendation (SiBraR). We define the notation and mathematically formulate the problem. Then, we describe how SiBraR leverages multimodal information of items and users for recommendation in warm- and cold-start scenarios. Finally, we describe a typical multi-branch architecture that we term multimodal **Mu**lti-**Bra**nch embedding network for **R**ecommendation (MuBraR), along with extensions that feature components of SiBraR.

3.1 Notation

Let $\mathcal{U} = \{u_i\}_{i=1}^M$ be the set of M users and $\mathcal{I} = \{i_j\}_{j=1}^N$ the set of N items. Moreover, let $R \in \mathbb{R}^{M \times N}$ be a binary matrix of interactions between users and items, where $r_{i,j} = 1$ indicates a *positive* interaction between user u_i and item i_j. $r_{i,j} = 0$ refers to an *unknown* interaction, as u_i might have consciously chosen not to interact with i_j, or just has not encountered it yet. In the rest of this work, and as common in literature [24, 48], we refer to items as *negatives* if their interactions are unknown. Let $\boldsymbol{u}_i \in \mathbb{R}^N$ be the i-th row of R, expressing user u_i's profile, i. e., a multi-hot vector encoding what items user u_i has interacted with. Analogously, let $\boldsymbol{i}_j \in \mathbb{R}^M$ be the j-th column of R, which expresses the profile of item i_j, i. e., all users that interacted with i_j.[1] Cold-start users (items) are those with an empty profile, i. e., they have no interactions in their profiles. We define the set of *modalities* (side information) associated and available for user u_i (e. g., gender and country) and item i_j (e. g., audio and text data) as $\mathcal{M}_i^{user}$ and $\mathcal{M}_j^{item}$, respectively. Moreover, we consider also interactions as a modality and therefore include non-empty profiles of users and items in the sets of modalities, i. e., $\{\boldsymbol{u}_i\} \subseteq \mathcal{M}_i^{user} \iff \|\boldsymbol{u}_i\| > 0$ and $\{\boldsymbol{i}_j\} \subseteq \mathcal{M}_j^{item} \iff \|\boldsymbol{i}_j\| > 0$, where $\|\cdot\|$ is the Euclidean norm.

3.2 SiBraR

We now describe SiBraR, which outputs a recommendation score $\hat{y}_{i,j}$ for a given user–item pair (u_i, i_j), by generating an embedding vector $\boldsymbol{e}_i$ for user u_i and an embedding vector for $\boldsymbol{e}_j$ for item i_j, and by computing their dot product $\hat{y}_{i,j} = \boldsymbol{e}_i \cdot \boldsymbol{e}_j$. The top k items of the list of descending $\hat{y}_{il}, l \in \{1, \ldots, N | r_{il} = 0\}$ that user u_i has not interacted with, form the recommendations for u_i. Assuming that different modalities of the same entity (i. e., user or item) share a similar set of underlying semantic information, SiBraR is designed to take any available modality as input, and to infer an embedding vector that lies in the same region as other modalities of the same entity in a shared embedding space [17]. Moreover, these vectors should be representative of the entity for providing accurate recommendations. To achieve this, SiBraR leverages weight-sharing of a deep neural network (DNN) g to embed different modalities. The network g is then optimized to provide an accurate representation for entities based on their available modalities taken as input. This encourages the network to map different modalities of the same user or item into the same shared embedding space. For instance, for item i_j, $g(m_1^i) \simeq g(m_2^i) \; \forall m_1^i, m_2^i \in \mathcal{M}_i^{item}$.

We now describe different variants of SiBraR. First, Item-SiBraR leverages multimodal item side information to tackle scenarios of item cold start and missing item modality. On the other hand, User-SiBraR utilizes multimodal user side information to tackle user cold start and missing user modality. Combining both approaches, User-Item-SiBraR tackles user and item cold start and missing modality.

---

[1]We use bold-face letters (e. g., $\boldsymbol{u}$ and $\boldsymbol{i}$) to denote vectors, while regular letters (e. g., u and i) for scalars and variables.
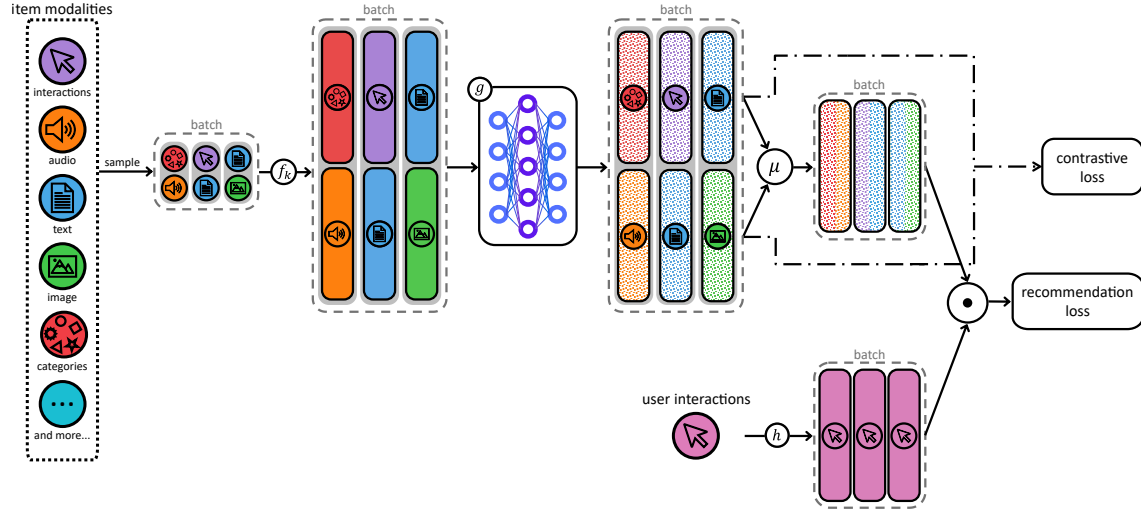
Fig. 1. `Item-SiBraR` architecture and its training procedure. `Item-SiBraR` consists of modality-specific item networks f_k, $k \in \{1, \ldots, N_{\text{mod}}\}$, the single-branch encoding network g shared across modalities and the user embedding network h. For each positive user–item interaction (u_i, i_j) in the training set, two modalities are sampled, independently forwarded through f_k and g before being averaged, indicated by μ, to generate a final representation for item i_j. The representation for user u_i is obtained through h. Recommendation scores are computed between positive items and the user representations. The recommendation loss ($\mathcal{L}_{\text{BPR}}$) considers the difference in scores for positive and negative items. On the other hand, the contrastive loss ($\mathcal{L}_{\text{SInfoNCE}}$) compares the similarity of different modalities of the same item to similarities of modalities to other items in the batch.

3.2.1 `Item-SiBraR`. Figure 1 shows the architecture and training pipeline of `Item-SiBraR`, and Algorithms 1 provides pseudocode for computing the batch loss employed in training. In `Item-SiBraR`, f_k, $k \in \{1, \ldots, |\mathcal{M}^{item}|\}$ denotes modality-specific functions, each expressed as a single linear layer and a ReLU activation function. They are used to transform different modalities to a common dimension required for the single-branch item embedding network g. If all modalities are already of the same shape, the networks f_k are unnecessary as g can be designed to take them as input as is. Thus, to ease notation, we omit them in the following. The item embedding network g, comprised of a series of linear layers, batch normalizations, and ReLU activation functions, forms a core component of `Item-SiBraR`. In contrast, for the user embedding network h, we consider either an embedding lookup table or an NN architecture similar to DeepMF [81]. For the lookup table, embeddings are randomly initialized and updated during training, while for the NN, the users' profiles are used as inputs to generate user embeddings.

During training, for a positive user–item pair, where user u_i has interacted with item i_j ($r_{i,j} = 1$), a set of N_{neg} non-interacted items $\mathcal{N}_i$, i.e., items the user u_i has not interacted with, are sampled uniformly at random. Moreover, for each pair, N_{mod} of all the item's modalities $\mathcal{M}_j^{item}$, are randomly sampled.[2] The process of modality sampling aims to improve robustness to missing modalities by reducing the dependence on any single input source. Each user embedding is obtained by $\boldsymbol{e}_i = h(u_i)$ or $\boldsymbol{e}_i = h(\boldsymbol{u}_i)$, depending on whether the lookup table or an NN is employed, respectively. The single-branch network g embeds the sampled modalities $\{m_1^j, \ldots, m_{N_{\text{mod}}}^j\} \subseteq \mathcal{M}_j^{item}$ of all positive and negative items alike, sharing weights across items and modalities. Finally, the embedding for item i_j is obtained by averaging the

---

[2]There exist different sampling techniques for negative sampling and modality sampling; we refer the reader to Yang et al. [82] and Saeed et al. [64], respectively. Among those, for simplicity we choose sampling uniformly at random, although we are aware that in real-world scenarios modalities are not uniformly available. We leave the application of other modality-sampling strategies for future work.

Algorithm 1: Pseudocode for computing the batch loss in `Item-SiBraR`

Input : $b \subset \{(u_i, i_j)|R_{ij} = 1\}$: Batch of user–item training interactions
    n_{neg}: Number of negatives per interaction
    $\mathcal{M}^{\text{item}}$: Item training modalities
    `ctrEmbs`: Whether to use contrastive loss or not
    λ, τ: Weight and temperature of contrastive loss

Output: $\mathcal{L}$: Loss for the batch

$\mathcal{L}_{\text{BPR}} \leftarrow 0$ // Reset BPR Loss
$\mathcal{L}_{\text{SInfoNCE}} \leftarrow 0$ // Reset contrastive loss
if `ctrEmbs` **then**
 | $n_{\text{mod}} \leftarrow 2$
else
 | $n_{\text{mod}} \leftarrow 1$
for $(u_i, i_j) \in b$ **do**
 | $\mathbf{e_i} \leftarrow h(u_i)$ // Embed user u_i
 | $\mathcal{N}_{ij} \leftarrow$ negSampling(u_i, i_j, R) // Sample negatives
 | $\mathcal{M}_{ij} \leftarrow$ randomSampling$(\mathcal{M}^{item})$ // Sample n_{mod} training modalities
 | $\mathbf{e_j} \leftarrow \frac{1}{n_{\text{mod}}} \sum_{m \in \mathcal{M}_{ij}} g(m_j)$ // Embed positive item i_j with single-branch g on modalities $\mathcal{M}_{ij}$
 | $\hat{y}_{ij} \leftarrow \mathbf{e_i} \cdot \mathbf{e_j}$ // Compute positive logit
 | **for** $i_k \in \mathcal{N}_{ij}$ **do**
  | $\mathbf{e_k} \leftarrow \frac{1}{n_{\text{mod}}} \sum_{m \in \mathcal{M}_{ij}} g(m_k)$ // Embed item i_k with single-branch g on modalities $\mathcal{M}_{ij}$
  | $\hat{y}_{ik} \leftarrow \mathbf{e_i} \cdot \mathbf{e_k}$ // Compute negative logit
  | $\mathcal{L}_{\text{BPR}} \leftarrow \mathcal{L}_{\text{BPR}} + \ln \sigma(\hat{y}_{ij} - \hat{y}_{ik})$ // Accrete BPR loss
 | **if** `ctrEmbs` **then**
  | $\mathcal{L}_{\text{SInfoNCE}} \leftarrow \mathcal{L}_{\text{SInfoNCE}} - \ln \frac{e^{(g(m_1^j) \cdot g(m_2^j))/\tau}}{\sum_{k \in \{j\} \cup N_{ij}} e^{(g(m_1^j) \cdot g(m_2^k))/\tau}} - \ln \frac{e^{(g(m_2^j) \cdot g(m_1^j))/\tau}}{\sum_{k \in \{j\} \cup N_{ij}} e^{(g(m_2^j) \cdot g(m_1^k))/\tau}}$
  | // Accrete contrastive loss
 | $\mathcal{L} \leftarrow \mathcal{L}_{\text{BPR}} + \lambda \mathcal{L}_{\text{SInfoNCE}}$

---

embeddings of its different modalities:

$$e_j = \frac{1}{N_{\text{mod}}} \left(g\left(m_1^j\right) + \ldots + g\left(m_{N_{\text{mod}}}^j\right) \right) \tag{1}$$

The recommendation scores are now computed by taking the scalar product between user and item embeddings, e_i and e_j, respectively. Employing Bayesian Personalized Ranking (BPR) [60], the recommendation loss can be computed based on these scores by

$$\mathcal{L}_{\text{BPR}}^{(u_i, i_j)} = - \sum_{k \in \mathcal{N}_i} \ln \sigma \left(\hat{y}_{i,j} - \hat{y}_{ik} \right), \tag{2}$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$ is the logistic sigmoid function.

While the effectiveness of sharing an embedding network to combine modalities in multimodal ML has been shown [63], we employ a contrastive loss to align embeddings from different modalities further. This loss is optional, and if it is not applied, $N_{\text{mod}} = 1$ and only a single modality is sampled per positive user–item pair. When the contrastive loss is applied, $N_{\text{mod}} = 2$ modalities per pair are sampled. In addition to the recommendation loss, `Item-SiBraR` adds the symmetric Noise Contrastive Estimation loss [55, 70] $\mathcal{L}_{\text{SInfoNCE}}$ as an additional loss term. $\mathcal{L}_{\text{SInfoNCE}}$ is computed between pairs of item modality embeddings of the positive and all the sampled negative items i_j and i_k, $k \in \mathcal{N}_i$ for

user u_i, respectively. It is defined as the symmetric version of the InfoNCE [55] contrastive loss,

$$\mathcal{L}_{\text{SInfoNCE}} = \mathcal{L}_{\text{InfoNCE}}^{1,2} + \mathcal{L}_{\text{InfoNCE}}^{2,1} \tag{3}$$

for the sampled modalities m_1 and m_2, and where

$$\mathcal{L}_{\text{InfoNCE}}^{1,2} = - \sum_{(u_i, i_j) \in \mathcal{B}} \ln \frac{e^{\left(g\left(m_1^j\right) \cdot g\left(m_2^j\right)\right)/\tau}}{\sum_{k \in \{j\} \cup \mathcal{N}_i} e^{\left(g\left(m_1^j\right) \cdot g\left(m_2^k\right)\right)/\tau}}. \tag{4}$$

The set $\mathcal{B}$ is a batch of randomly sampled user–item pairs. Moreover, the parameter τ controls the expression's contrastive temperature and is considered a hyperparameter in the experiments conducted in Section 4. Intuitively, $\mathcal{L}_{\text{SInfoNCE}}$ enforces similarity between different modality embeddings of the same item whereas dissimilarity between modality embeddings of different items. The full loss formulation is now given by

$$\mathcal{L} = \mathcal{L}_{\text{BPR}} + \alpha \mathcal{L}_{\text{SInfoNCE}}, \tag{5}$$

where α controls the weight of the contrastive loss term.

During inference, `Item-SiBraR` utilizes all available item modalities $\mathcal{M}_j^{item}$ for any item i_j, embedding them with the single-branch network g and aggregating the resulting embeddings:

$$e_j = \frac{1}{|\mathcal{M}_j^{item}|} \sum_{m^j \in \mathcal{M}_j^{item}} g(m^j) \tag{6}$$

In case of item cold-start, the empty item profile i_j is not considered, i. e., $i_j \notin \mathcal{M}_j^{item}$, and the resulting embedding is obtained using only item side information. During inference, a single modality, i. e., $|\mathcal{M}_j^{item}| \geq 1$ is already sufficient to provide recommendations.

3.2.2 `User-SiBraR`. To mitigate the user cold-start and missing modality problems, i. e., when a user's profile u_i or any additional side information is missing, `User-SiBraR` can be employed. Analogous to `Item-SiBraR`, h now produces the item embeddings e_j either from a learnable lookup table or by embedding the item's profile i_j via an NN. The modality-specific user networks f_k, $k \in \{1, \ldots, |\mathcal{M}^{user}|\}$, simple one-layer NNs, transform individual user modalities to a common space for further usage. Finally, the single-branch user embedding module g, an NN with several linear layers, batch normalizations, and ReLU activation functions, embeds all user modalities into a shared embedding space. By minimizing the full loss (Equation 5), `User-SiBraR` enforces similarity between modality embeddings of the same user while dissimilarity between modality embeddings of different users. During inference, `User-SiBraR` generates the embedding for u_i by averaging all available user modality embeddings forwarded through the modality-specific networks f_k and the single-branch g. We again omit the former for simplicity.

$$e_i = \frac{1}{|\mathcal{M}_i^{user}|} \sum_{m^i \in \mathcal{M}_i^{user}} g(m^i) \tag{7}$$

3.2.3 `User-Item-SiBraR`. For completeness, by combining `User-SiBraR` and `Item-SiBraR` the cold-start and missing modality problems can be mitigated for users and items simultaneously. We term this network `User-Item-SiBraR`. We denote the modality-specific entity networks as f_k^{user}, $k \in \{1, \ldots, |\mathcal{M}^{user}|\}$ and f_k^{item}, $k \in \{1, \ldots, |\mathcal{M}^{item}|\}$, for users and items, respectively. Moreover, `User-Item-SiBraR` features two single-branch networks g^{user} and g^{item}. The
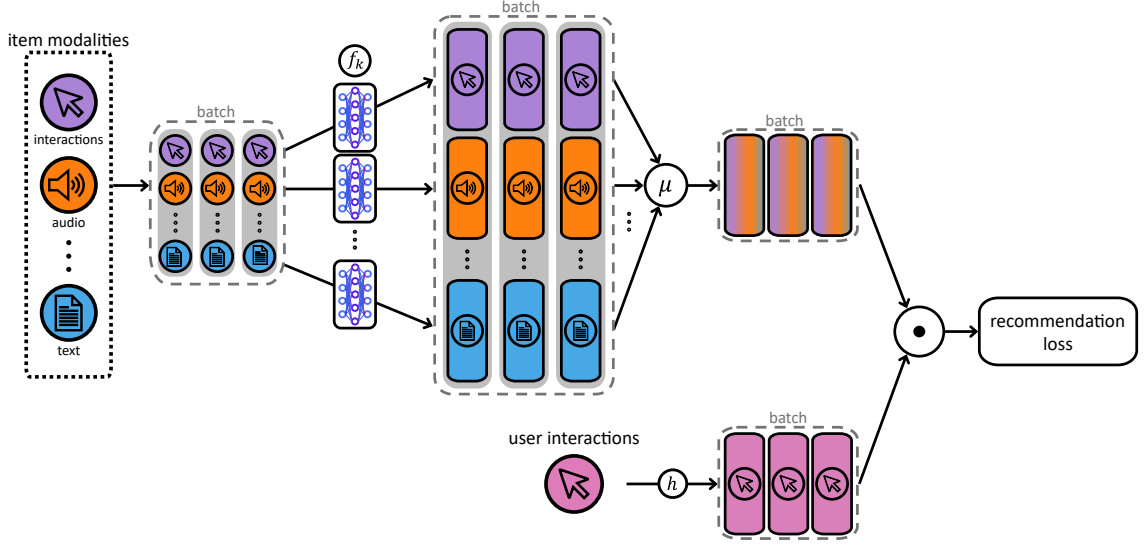
Fig. 2. MuBraR model and training procedure. The MuBraR network represents the multi-branch embedding network, and each branch f_k (a DNN), is only used by a single modality. A final item representation is obtained by taking an average over all available modality representations. The representation for a user is obtained through h. For each user–item interaction pair (u_i, i_j) in the training set, the recommendation score is computed by the dot product between user and item representations. Moreover, the recommendation loss ($\mathcal{L}_{\text{BPR}}$) is computed between the recommendation scores of positive and negative items.

training is done by minimizing the loss formulation

$$\mathcal{L} = \mathcal{L}_{\text{BPR}} + \alpha \mathcal{L}^{user}_{\text{SInfoNCE}} + \beta \mathcal{L}^{item}_{\text{SInfoNCE}} \tag{8}$$

where $\mathcal{L}^{user}_{\text{SInfoNCE}}$ and $\mathcal{L}^{item}_{\text{SInfoNCE}}$ are the contrastive losses between user and item modalities, respectively. α and β are hyperparameters that control the influence of these terms.

3.3 MuBraR

For comparison of SiBraR with standard multimodal NNs found in the literature, we introduce MuBraR. It captures the core concept of having an individual embedding network, typically a DNN, per modality, that many works use [11, 25, 72, 77]. MuBraR outputs a recommendation score $\hat{y}_{i,j}$ for each given user–item pair (u_i, i_j) based on user and item embeddings e_i and e_u, and taking their dot product $\hat{y}_{i,j} = e_i \cdot e_j$. Like SiBraR, the top k items a user has not interacted with are recommended to them. We now describe Item-MuBraR, a variant of MuBraR focusing on mitigating the item cold-start and missing modality problem. As User-MuBraR, the user modality-based version, follows accordingly, we will omit its description and refer to Section 3.2.2 instead.

3.3.1 Item-MuBraR. Figure 2 shows the architecture and training pipeline of Item-MuBraR. For user u_i, its embedding e_i is obtained using either an embedding lookup table or embedding the user profile via a DNN. For item i_j, Item-MuBraR embeds each available item modality $m^j_k \in \mathcal{M}^{item}_j$ via a modality-specific network (or branch) f_k, which are DNNs several layers deep. As the aim of this study is to provide an in-depth understanding of SiBraR, we accordingly choose

to fuse the individual item modality embeddings by averaging:

$$e_j = \frac{1}{|\mathcal{M}_j^{item}|} \sum_{k=1}^{|\mathcal{M}_j^{item}|} f_k(m_k^j) \tag{9}$$

This aligns more closely with the core concept of SiBraR and thus facilitates an fair comparison between both architectures. Moreover, it allows for seamless handling of missing modalities by excluding their embeddings from the averaging process. Despite these benefits to our study, we also acknowledge the existence of various other strategies for fusing modality embeddings, e. g., embedding concatenation (with or without subsequent processing through a NN) as seen in [11, 25, 72]. As such, we select DropoutNet as an additional baseline to compare SiBraR against (see Section 4.3).

Considering that SiBraR utilizes modality sampling and a contrastive loss for modality embeddings, both of which require further investigation regarding their impact on the recommendation task, we propose two additional variants of MuBraR: (1) MuBraR$_S$ adds modality sampling and (2) MuBraR$_{SC}$ further extends this idea by adding a contrastive loss term during training. Doing so, these two extensions applied on MuBraR (as an architecture without weight-sharing) allows us to gather detailed insights on the utility and effects of SiBraR's three main training paradigms.

The modified architecture and training pipeline are shown in Figure 3. In comparison to Item-SiBraR (Figure 1), there is no single-branch network, while the modality-specific layers feature learning capabilities through more linear layers coupled with ReLU activation functions.

The main and only differences to Item-SiBraR are the missing single-branch network and the stronger learning capabilities of the modality-specific layers. Hence, much of the training procedure follows accordingly. For MuBraR$_S$, when sampling modalities, $N_{\mathrm{mod}} = 1$ and $e_j = f_1(m_1)$, where $m_1 \in \mathcal{M}_j^{item}$ is the sampled modality, and f_1 is the corresponding modality network. The training loss is simply the BPR loss, as stated in Equation 2. For MuBraR$_{SC}$, when utilizing a contrastive loss on the modality representations, we have $N_{\mathrm{mod}} = 2$ and

$$e_j = \frac{1}{2}\left(f_1(m_1) + f_2(m_2)\right) \tag{10}$$

for the sampled modalities $\{m_1, m_2\} \subseteq \mathcal{M}_j^{item}$ and their respective networks f_1 and f_2. The training loss is again computed as in Equation 5 and Equation 3, whereas $\mathcal{L}_{\mathrm{InfoNCE}}^{1,2}$ is now given by

$$\mathcal{L}_{\mathrm{InfoNCE}}^{1,2} = - \sum_{(u_i, i_j) \in \mathcal{B}} \ln \frac{e^{\left(f_1\left(m_1^j\right) \cdot f_2\left(m_2^j\right)\right)/\tau}}{\sum_{k \in \{j\} \cup \mathcal{N}_i} e^{\left(f_1\left(m_1^j\right) \cdot f_2\left(m_2^k\right)\right)/\tau}}. \tag{11}$$

During inference, all available item modalities $\mathcal{M}_j^{item}$ are used to generate an item embedding e_j (Equation 9), and a dot product is calculated with user embeddings e_i to obtain the recommendation score for user–item pair (u_i, i_j).

4 Experimental Setup

In this section, we describe the experimental setup for our experiments, including the datasets, the baselines used for comparison, the evaluation protocol, the training procedure, and hyperparameter tuning. Moreover, we will publicly release the code used to carry out our experiments.[3]
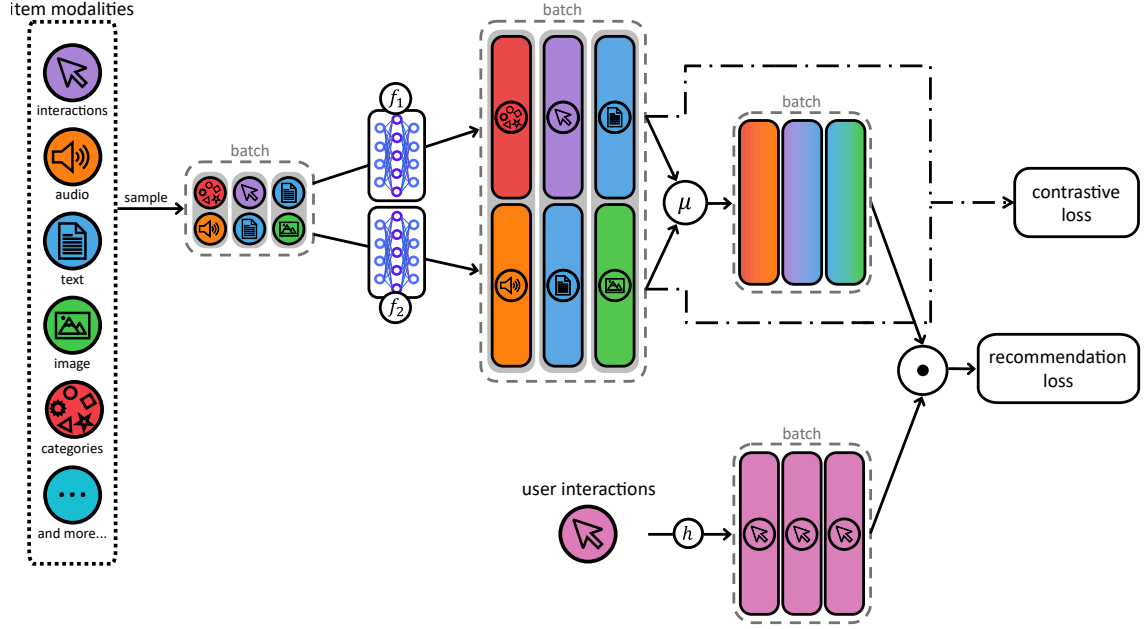
Fig. 3. MuBraR$_{SC}$ (MuBraR$_{SC}$) architecture and its training procedure. The network consists of modality-specific item networks f_k, $k \in \{1, \ldots, N_{mod}\}$ and the user embedding network h. For each positive user–item interaction (u_i, i_j) in the training set, two modalities are sampled (one for MuBraR$_S$), independently forwarded through the specific networks f_k, before being averaged, indicated by μ, to generate a final representation for item i_j. The representation e_i for user u_i is obtained through h. Recommendation scores are computed between positive items and the user representations. The recommendation loss ($\mathcal{L}_{BPR}$) considers the difference in scores for positive and negative items. On the other hand, the contrastive loss ($\mathcal{L}_{SInfoNCE}$) compares the similarity of different modalities of the same item to similarities of modalities to other items in the batch.

					User Features				Item Features			
	# users	# items	# inter.	sparsity	Age (**d**)	Gender (**c**)	Country (**c**)	Occupation (**c**)	Text (**v**)	Audio (**v**)	Image (**v**)	Genre (**m**)
Onion [43]	5,192	13,610	407,653	0.99423	–	2	152	–	768	50; 100; 4,800	4,096	853
ML-1M [19]	5,816	3,299	813,792	0.95759	7	2	–	21	768	–	–	18
Amazon [22]	11,454	4,177	87,098	0.99818	–	–	–	–	768; 768	–	2,048	–

Table 1. This table provides the summary of datasets. Left side: Characteristics after pre-processing. Right side: Representation type (**c**ategorical, **d**iscrete, **v**ector, **m**ultilabel), dimensionality (for **v**) or number of options (for **c**, **d**, and **m**) of each content feature available. Dimensionalities separated by a colon indicate that more than one feature is available; features that are not available or have been neglected are denoted by – (see Section 4.1).

4.1 Datasets

We use the same dataset selection, preprocessing, and data filtering procedure as described in Ganhör et al. [17]. We conduct our experiments in three different domains: music, movie, and e-commerce. We carefully selected the datasets by considering their popularity for benchmarking RSs, their number of content features, and the multimodality of available side information.

This work concentrates on the implicit feedback scenario. We therefore construct a binary user–item interaction matrix R: $R_{ij} = 1$ if user u_i interacted with item i_j and $R_{ij} = 0$ otherwise.

Table 1 outlines the traits of the datasets after pre-processing (further explained below). The table also gives insights into the modalities and dimensionalities of the side information used in our experiments.

4.1.1 *Music4All-Onion.* (Onion) [4] [43]

This large-scale multimodal music dataset provides three types of information: 1) user–item interactions, 2) side information on users, and 3) side information on items. As side information on users, we select their gender and country. The dataset provides a wide range of item features, including several representations linked to the music tracks' audio signals, lyrics, and YouTube video clips. From these, we select those pertaining to the audio signal in terms of i-vectors [15] and the visual representations of the YouTube video clips extracted with a pre-trained instance of ResNet [20] for our experiments.

As in [17], we extend the available representations for audio and lyrics by applying more recent NN architectures for music information retrieval and natural language processing to the provided data. We select the NNs MusicNN[5] [51, 52] and Jukebox[6] [14] to extract representations of the 30s-audio-snippets. After pre-processing the lyrics as described in [43], we encode the text using all-mpnet-v2[7] pre-trained instance of the SentenceTransformer model [58] provided by HuggingFace [78].

We only consider users of age between 10 and 80 and users and items with all side information available. We further follow [17] by creating a subset of listening events over the year 2018 for our experiments to follow a standard practice in the music RS domain [40, 41].

Finally, we transform the interactions to binary implicit feedback, similar to [16, 40, 42]. We use a threshold of 2 on the user–item interaction counts, i. e., a user must have listened to an item at least twice, and apply *5-core filtering* for users and items, which only keeps users with at least 5 interactions and items that have been interacted with by at least 5 users.

4.1.2 *MovieLens 1M.* (ML-1M) [8] [19]

This dataset provides user ratings of movies gathered from the MovieLens website.[9] The dataset consists of user–item interactions, side information on users (age, gender, and occupation) and on items (movie genre).

As in Ganhör et al. [17], we enhance the dataset by crawling Wikipedia for movie plots and further encode these texts with the all-mpnet-v2[10] pre-trained instance of the SentenceTransformer model [58] provided by HuggingFace [78].

We filter the data to only include users between 10 and 80 years old and those users and items for which all features are available in the dataset. We transform the rating information into binary implicit feedback by keeping only the interaction information of ratings >= 3. Finally, we apply 5-core filtering on users and items alike.

---

[4] https://zenodo.org/records/6609677
[5] https://github.com/jordipons/musicnn/tree/master
[6] https://github.com/openai/jukebox/
[7] https://huggingface.co/sentence-transformers/all-mpnet-base-v2
[8] https://grouplens.org/datasets/movielens/1m/
[9] https://movielens.org/
[10] https://huggingface.co/sentence-transformers/all-mpnet-base-v2

users	item interactions
user 1	3, 1, 32, 8, 18, 10, 7, 37, 10, 21
user 2	2, 25, 24, 31, 19, 7
user 3	1, 39, 15, 9, 19, 31, 22, 3, 2, 6
. . .	. . .
user n	38, 9, 11, 28, 22, 10, 27, 3, 23, 33, 27

Fig. 4. Outline on a warm-start dataset scenario. The item interactions of each user in the dataset are split randomly into train, validation, and test set, as indicated by the different colors. Each user and item is present in the train set, and may be in the validation and test sets.

4.1.3   Amazon Video Games. (Amazon) [11] [22] Amazon Reviews'23 is a large-scale dataset which provides Amazon reviews dated from May 1996 to September 2023. This dataset is an expanded version of the well-established Amazon Reviews'18 dataset [47].

Amazon provides various item features, including the title, description, price, and at least one product image link per item. In this work, we chose to take the category *Video Games* as a subset of the Amazon dataset for our experiments. Following [17], we decided on this subset by carefully considering multiple factors, such that the resulting subset consists of a large enough set of users, items, and user–item interactions to replicate a real-world scenario, and provides unique features considering our other selected datasets. We filter the reviews to be dated between January 2016 and December 2019. Moreover, we keep only items with available titles, descriptions, and at least one high-resolution image per product. We consider the ratings as implicit feedback with a threshold of 1 and apply 5-core filtering. As in [17], we use `all-mpnet-v2`[12] pre-trained instance of the SentenceTransformer model [58] provided by HuggingFace [78] to encode the texts (product title and description) of this dataset. After downloading the first high-resolution image of the product, we preprocess it following the same approach as Saeed et al. [63]. First, the image is rescaled such that the smaller edge has a size of 256. The resulting image is then cropped at the center to a square of 224 pixels. Each channel is then normalized to the mean and standard deviation of ImageNet [13]. We then compute the encoding of the final image with the pre-trained instance of `ResNet-101` [20] provided by TorchVision[13] [39].

4.2   Data Scenarios

To assess the performance of the RSs in different scenarios, we split the datasets in three different ways, each featuring a train, validation, and test set. While models are trained on the train set, hyperparameter optimization is done on the validation set. The test set is only used to report the final results, estimating a model's performance on unseen data.

Warm start (random split). We randomly select 80%, 10%, and 10% of each user's interactions for training, validation, and testing, respectively. Figure 4 illustrates this scenario.

User cold start. We randomly split 80%, 10%, and 10% of all the users in the dataset into disjoint sets of train, validation, and test users, respectively. RSs are trained on all user interactions in the train set while ignoring those in validation and test sets. During validation (testing), only validation (test) users are considered, while train and test (validation) users are disregarded. Since test users have not been seen during model training and validation/model selection, this

---

[11]https://amazon-reviews-2023.github.io/index.html

[12]https://huggingface.co/sentence-transformers/all-mpnet-base-v2

[13]https://pytorch.org/vision/main/models/generated/torchvision.models.resnet101.html
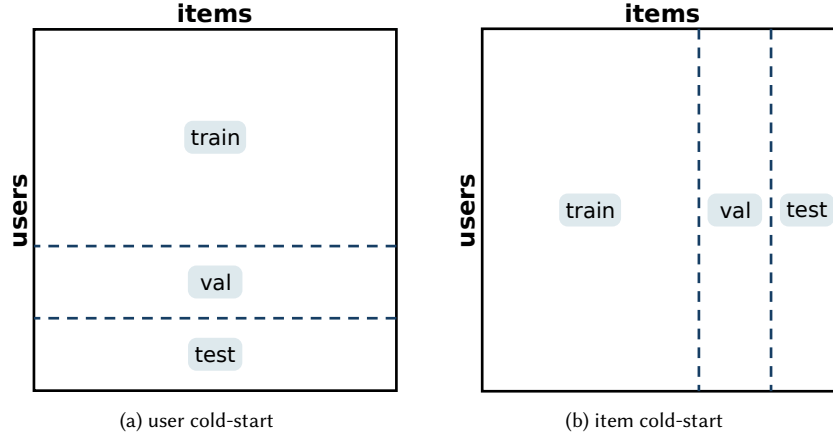
Fig. 5. Outline on user and item cold-start scenario. The whole square represents all users and items of a dataset, which can be thought of as the interaction matrix containing user–item interaction pairs. It is split into train, validation, and test sets, as indicated by the different subregions labeled *train*, *val*, and *test*, respectively. In (a), users are split randomly such that no user exists in more than one set. Analogously, in (b), no item exists in more than one set.

split is suitable for assessing the performance of RSs in user cold-start scenarios. Figure 5a visually explains this user cold-start scenario.

Item cold start. We randomly split 80%, 10%, and 10% of all items in the dataset into disjoint sets of train, validation, and test items, respectively. RSs are trained on all item interactions in the train set while ignoring those in validation and test sets. During validation (testing), only validation (test) items are considered, while train and test (validation) items are disregarded. Since test items have not been seen during model training and validation/model selection, this split is suitable for assessing the performance of RSs in item cold-start scenarios. Figure 5b visually explains this item cold-start scenario.

4.3 Baselines

We compare SiBraR with two HRSs for cold-start, CLCRec [77] and DropoutNet [72], which are often used as baselines addressing cold-start scenarios, leveraging CF signals and multimodal side information. Moreover, they employ both explicit (CLCRec) and implicit (DropoutNet) approaches for multimodal learning (see Section 2). Additionally, we include two CF models, matrix factorization (MF) with BPR [60] and Deep Matrix Factorization (DeepMF) [81]. We select the former based on its simplicity and yet effectiveness in recommendations tasks by factorizing user and item latent representations. DeepMF features a two-tower DNN with user and item interactions as inputs. To compare different single- and multi-branch multimodal networks, we additionally consider the variants described in Section 3. We further employ two naive baselines recommender systems (BRSs): Rand, which, for each user, randomly samples items to recommend, and Pop, which recommends to all users the same most popular items – as described by the number of individual user interactions in the training set.

4.4 Metrics

We evaluate the performance of our algorithms by accuracy and beyond-accuracy metrics. While the former estimates whether users would consume recommended items, the latter measures aspects beyond consumption, which can, among others, indicate possible biases the algorithms might exhibit. Moreover, we measure the *modality gap* [23, 31] by two metrics between the modalities of an item and between different items.

Let $\mathrm{Rel}(u_i) \subset I$ be the list of relevant items for user u_i. When evaluating and testing an algorithm, these are the items in the respective validation or test split. $\mathrm{Rec}(u_i) \subset I, k = |\mathrm{Rec}(u_i)|$ is the ordered list of top-k items recommended to the user, and $\mathrm{Rec}(u_i)_j$ selects the j-th item of this list. $\mathbf{1}_{\mathrm{Rel}(u_i)}(i_j)$ indicates that item i_j is present in $\mathrm{Rel}(u_i)$, more formally

$$\mathbf{1}_{\mathrm{Rel}(u_i)}(i_j) := \begin{cases} 1 & \text{if } i_j \in \mathrm{Rel}(u_i), \\ 0 & \text{if } i_j \notin \mathrm{Rel}(u_i). \end{cases}$$

In the following, we define the metrics on a user level, noting that only averages over all users are reported later on. For accuracy and beyond-accuracy metrics, where applicable, we test the best algorithms for significant improvement over the other algorithms using multiple paired t-tests, with Bonferroni correction to account for the multiple comparisons. We consider the improvement statistically significant if $p < 0.05$.

4.4.1 Accuracy metrics. We first describe NDCG, our primary measure of recommendation quality, followed by additional metrics.

Normalized Discounted Cumulative Gain (NDCG) evaluates the ranking quality of an RS. In short, it weighs good recommendations, i. e., ones that a user likes, stronger the earlier in the list of recommended items they occur. When computed on top-k recommendations, NDCG@k is given by

$$\mathrm{NDCG}@k(u_i) = \frac{\mathrm{DCG}@k(u_i)}{\mathrm{IDCG}@k(u_i)},$$

where DCG@k is the Discounted Cumulative Gain (DCG)

$$\mathrm{DCG}@k(u_i) = \sum_{j=1}^{k} \frac{\mathbf{1}_{\mathrm{Rel}(u_i)}(\mathrm{Rec}(u_i)_j)}{\log_2(j+1)}$$

and IDCG@k is the Ideal Discounted Cumulative Gain (IDCG), when all recommended items are relevant, i. e., $\mathrm{IDCG}@k(u_i) = \sum_{j=1}^{k} \frac{1}{\log_2(j+1)}$ if there are k relevant items for user u_i.

Recall and Precision for user u_i only consider the presence of a relevant item in the recommendations, ignoring the order of items in the list:

$$\mathrm{Recall}@k(u_i) = \frac{|\mathrm{Rel}(u_i) \cap \mathrm{Rec}(u_i)|}{|\mathrm{Rel}(u_i)|} \qquad \text{and} \qquad \mathrm{Precision}@k(u_i) = \frac{|\mathrm{Rel}(u_i) \cap \mathrm{Rec}(u_i)|}{|\mathrm{Rec}(u_i)|}.$$

F-score@k applies a harmonic mean on Recall@k and Precision@k to represent both metrics at once:

$$\mathrm{F\text{-}score}@k(u_i) = 2 \frac{\mathrm{Precision}@k(u_i) \cdot \mathrm{Recall}@k(u_i)}{\mathrm{Precision}@k(u_i) + \mathrm{Recall}@k(u_i)}.$$

Average Precision (AP), also known as *Mean Average Precision* (MAP) when averaged over all users, considers the order of recommended items by evaluating Precision@k at multiple levels of k.

$$\mathrm{AP}@k(u_i) = \frac{\sum_{j=1}^{k} \mathbf{1}_{\mathrm{Rel}(u_i)}(\mathrm{Rec}(u_i)_j) \cdot \mathrm{Precision}@j(u_i)}{|\mathrm{Rel}(u_i)|}$$

Reciprocal Rank (RR) [73], also known as *Mean Reciprocal Rank* (MRR) when averaged over all users, assumes that users are satisfied after having encountered the first relevant item recommended at a high rank.

$$\text{RR@}k(u_i) = \frac{1}{min_k\{k; \text{Rec}(u_i)_k \in \text{Rel}(u_i)\}}$$

4.4.2 Beyond-accuracy metrics. A wide variety of metrics exist beyond the common accuracy-based model performance indicators above. In this work, we focus on four metrics that can shed light on the popularity bias of RSs, which describes how RSs might recommend already popular items more often while scarcely or never recommending low popular items. This is problematic due to its adverse effects on many different stakeholders, e. g., users may not find items of interest or content providers may miss sales opportunities if their items are not recommended. For a survey on popularity bias, we refer the reader to [27].

Coverage indicates what proportion of items of the whole catalog are recommended during evaluation.

$$\text{Coverage@}k = \frac{|\bigcup_{u_i \in \mathcal{U}} \text{Rec}(u_i)|}{|\mathcal{I}|}$$

Average recommendation popularity (ARP) [26] investigates the average popularity of items recommended to the users. While the popularity $\phi(i_j)$ of an item i_j can be measured in different ways, we use $\phi(i_j) = \frac{|\mathcal{U}_j|}{|\mathcal{U}|}$ with $\mathcal{U}_j$ indicating training set users that interacted with the item [44].

$$\text{ARP@}k(u_i) = \frac{1}{|\text{Rec}(u_i)|} \sum_{i_j \in \text{Rec}(u_i)} \phi(i_j)$$

Average Percentage of Long Tail Items (APLT) [1, 26] assesses the proportion of long tail items Γ, i. e., set of items with the least popularity in the recommendations. We consider 80% of the least popular items as the long tail.

$$\text{APLT@}k(u_i) = \frac{|\text{Rec}(u_i) \cap \Gamma|}{|\text{Rec}(u_i)|}$$

Popularity lift (PL) [2] measures how RSs distort the popularity bias of users inherent in their histories.

$$\text{PL@}k(u_i) = \frac{\text{PB}_q(u_i) - \text{PB}_p(u_i)}{\text{PB}_p(u_i)},$$

where $\text{PB}_p(u_i) = \frac{1}{|\text{Rec}(u_i)|} \sum_{i_j \in \text{Rec}(u_i)} \phi(i_j)$ and $\text{PB}_q(u_i) = \frac{1}{|\text{Rel}(u_i)|} \sum_{i_j \in \text{Rel}(u_i)} \phi(i_j)$ measure the average item popularity of user u_i's history and recommendations, respectively.

4.4.3 Describing the embedding space. We provide quantitative measures to describe the modality gap to gain insights on the shared embedding space of multimodal models. This gap is linked to a model's performance on downstream tasks [31].

Euclidean distance (ED) and *cosine similarity* (CS) measure the distance and similarity between two d-dimensional vectors $a \in \text{R}^d$ and $b \in \text{R}^d$, respectively.

$$\text{ED}(a, b) = \|a - b\| \qquad \text{and} \qquad \text{CS}(a, b) = \frac{a \cdot b}{\|a\| \, \|b\|},$$

where $\|\cdot\|$ is the Frobenius norm.

Inter-item metrics measure the average difference in the modality embeddings of the same type between different items. Let $\mathcal{M}$ be the set of N_{mod} unique modalities of all N items. We define $e_j^{m_k} \in \text{R}^d$ to be the embedding for the modality $m_k \in \mathcal{M}$ of item $i_j \in \mathcal{I}$. The inter-item metric is then given by

$$\text{inter}_f = \frac{1}{N_{\text{mod}}} \frac{2}{N(N-1)} \sum_{k=1}^{N_{\text{mod}}} \sum_{i=1}^{N} \sum_{j=1}^{i-1} f(e_i^{m_k}, e_j^{m_k}),$$

where $f(\cdot, \cdot)$ is either ED or CS. Based on the symmetry of f, we only compute it once per pair. Moreover, we neglect measuring embeddings to themselves. A low inter_{ED} expresses close proximity of the same modalities of different items while a low inter_{CS} indicates strong dissimilarity between the modalities.

Analogously, *intra-item metrics*, which quantify the average difference between modalities of the same item, are defined as

$$\text{intra}_f = \frac{1}{N} \frac{2}{N_{\text{mod}}(N_{\text{mod}}-1)} \sum_{j=1}^{N} \sum_{k=1}^{N_{\text{mod}}} \sum_{l=1}^{k-1} f(e_j^{m_k}, e_j^{m_l}).$$

4.5 Hyperparameter Tuning

We perform an extensive hyperparameter search on `SiBraR` and all other algorithms reported in Section 4.3. Where applicable, we tune their learning rates, weight decays, dropout rates, model dimensions, and number of layers. For explicit HRSs, i.e., those with additional loss terms, we search over possible loss parameters, including contrastive weights and temperatures for `SiBraR` and `MuBraR`. Moreover, for HRSs, we treat training modalities as another hyperparameter. For DropoutNet, `SiBraR`, and `MuBraR`, which are not restricted by the number of modalities, we try out all possible combinations of various sizes (1 up to N_{mod}). To make a fair comparison with CLCRec, which supports only one modality besides interactions, we additionally tune all HRSs on only sets consisting of interaction and another modality. We denote the best-performing HRSs in this setting by DropoutNet$_{\text{one}}$, `SiBraR`$_{\text{one}}$ and `MuBraR`$_{\text{one}}$, while their best-performing counterparts leveraging any number of modalities as DropoutNet$_{\text{best}}$, `SiBraR`$_{\text{best}}$ and `MuBraR`$_{\text{best}}$, respectively. All algorithms are trained for up to 50 training epochs. For algorithms requiring negative samples, we sample 10 for each positive user-item pair. For computing the metrics (Section 4.4), we use the cut-off threshold $k = 10$. After each training cycle, the algorithms are evaluated on the validation set and the NDCG@10 is computed and used as validation metric for hyperparameter selection. We perform early-stopping by continuously storing the best model over the epochs and stopping the training when no further NDCG@10 improvements can be observed for five consecutive epochs. Only the models achieving the highest NDCG@10 on the validation set are evaluated on the test set, ensuring no information flow from testing to model training. We exclusively use data present in the test sets when reporting the outcomes of our experiments (Section 5). We utilize the Weights and Biases (W&B) platform [7][14] for monitoring and the Bayesian optimization of W&B Sweeps for tuning the hyperparameters. For an overview of all experiments performed and ranges of hyperparameters employed, we refer the reader to Table 10 in the appendix as well as the the configuration files available in our repository.[15]

5 Results

In this section, we first compare the performance of `SiBraR` to standard CFs and HRSs, an then analyze different single- and multi-branch variants and their components. Additionally, we study how contrastive loss parameters affect single-branch networks, and conclude by investigating the modality gap of multimodal RSs.

---

[14]https://wandb.ai/
[15]https://github.com/hcai-mms/single-branch-networks

		Warm-Start			Item Cold-Start			User Cold-Start	
		ML-1M	Onion	Amazon	ML-1M	Onion	Amazon	ML-1M	Onion
BRS	Rand	0.0045	0.0009	0.0012	0.0497	0.0074	0.0118	0.0445	0.0074
	Pop	0.1268	0.0182	0.0255	0.0087	0.0063	0.0083	0.4583	0.0994
CF	MF	0.2589	0.1438	**0.0818**†	0.0300	0.0055	0.0125	0.4613	0.0920
	DeepMF	0.1165	0.1297	0.0556	0.0087	0.0063	0.0083	0.4124	0.0575
HRS	CLCRec	**0.2680**†	0.1378	0.0569	0.1762	0.1503	0.1348	0.4616	0.0987
	DropoutNet$_{one}$	0.2510	0.1078	0.0413	0.2000	0.1221	0.1033	0.4582	0.0947
	MuBraR$_{one}$	0.2603	0.1471	0.0535	0.2751	0.2289	0.1262	0.4553	0.0946
	SiBraR$_{one}$	0.2593	0.1505	0.0728	0.2592	0.2183	**0.1479**†	**0.4673**	0.0918
	DropoutNet$_{best}$	0.2510	0.1078	0.0455	0.2000	0.1689	0.1033	0.4582	0.1086
	MuBraR$_{best}$	0.2603	**0.1762**	0.0535	0.2949	0.2276	0.1262	0.4501	0.0951
	SiBraR$_{best}$	0.2581	0.1717	0.0728	**0.2994**	**0.2297**	**0.1479**†	0.4659	**0.1125**

Table 2. NDCG@10 results for different BRSs, CFs, and HRSs. Each row indicates the results of the evaluated algorithm, and the columns blocks indicate the data scenario (warm start and user/item cold start) for the datasets (ML-1M, Onion, and Amazon). Bold indicates the best-performing RS, † indicates significant improvement over all others (paired t-tests with $p < 0.05$ considering Bonferroni correction). The results of the different scenarios are not comparable due to the different numbers of users and items in the test splits.

5.1 Baseline comparison

The primary aim of this section is to answer RQ1 (see Section 1). In the following, we compare SiBraR with basic, collaborative filtering, and hybrid recommenders, including multi-branch variants.

5.1.1 Accuracy. Table 2 shows the NDCG@10 test scores of all algorithms on different datasets, in warm-start as well as user and item cold-start scenarios. Considering the lack of user side information of the Amazon dataset, for this dataset algorithms have only been trained on warm and item cold start, while they have been trained on all scenarios for ML-1M and Onion. Each row indicates the NDCG@10 test results of a specific algorithm (see Section 4.3), which are further grouped by the horizontal solid lines into basic (BRS), collaborative filtering-based (CF), and hybrid (HRS) recommender systems. For brevity, we jointly report the best variant of Item-SiBraR and User-SiBraR as SiBraR, and of Item-MuBraR and User-MuBraR as MuBraR. Moreover, for algorithms that allow more than one modality in addition to interactions, i. e., DropoutNet, MuBraR, and SiBraR, for a fair comparison with CLCRec, we list the models leveraging only a single side-modality with the suffix *one*, e. g., SiBraR$_{one}$, and those that report the best performance for any number of modalities with the suffix *best*, e. g., SiBraR$_{best}$. The table further indicates this by the dashed line separating these HRSs. We refer the reader to the appendix for results on the other accuracy metrics mentioned in Section 4.4.1.

Warm start. In warm start, i. e., where user and item interaction histories are also leveraged for recommendation, MF, CLCRec, MuBraR, and SiBraR are the four top-performing algorithms. The order varies with the datasets: CLCRec performs best on ML-1M, MuBraR on Onion, and MF on Amazon. Considering only algorithms with at most one modality besides interactions, CLCRec achieves a top performance on ML-1M, SiBraR on Onion, and MF on Amazon.

Cold start. The weakness of BRSs and CFs is evident when considering cold-start scenarios, where HRSs outperform them. While a more considerable improvement over CF techniques is observable for item than for user cold-start, we

		ML-1M			Onion			Amazon		
		ARP↓	APLT↑	PL→	ARP↓	APLT↑	PL→	ARP↓	APLT↑	PL→
BRS	Rand	**0.0307**†	**0.8206**†	-0.7257	**0.0045**†	**0.8024**†	-0.5694	**0.0013**†	**0.8005**†	-0.5016
	Pop	0.3273	0.0000	1.9425	0.0759	0.0000	6.2455	0.0249	0.0000	8.7718
CF	MF	0.1690	0.1031	0.4158	0.0213	0.1641	0.6614	0.0100	0.1345	1.5492
	DeepMF	0.1174	0.0679	**-0.0295**†	0.0199	0.1266	0.5353	0.0053	0.1976	**0.2144**†
HRS	CLCRec	0.1698	0.1049	0.4359	0.0227	0.1377	0.8065	0.0069	0.2241	0.9262
	DropoutNet_{one}	0.1825	0.0845	0.5395	0.0210	0.2143	0.6489	0.0099	0.1441	1.8784
	MuBraR_{one}	0.1742	0.0926	0.4584	0.0167	0.2385	0.3137	0.0081	0.2159	0.8433
	SiBraR_{one}	0.1911	0.0846	0.6082	0.0155	0.3126	**0.1942**†	0.0085	0.1522	1.0296
	DropoutNet_{best}	0.1825	0.0845	0.5395	0.0210	0.2143	0.6489	0.0126	0.1178	2.4068
	MuBraR_{best}	0.1742	0.0926	0.4584	0.0203	0.1871	0.5870	0.0081	0.2159	0.8433
	SiBraR_{best}	0.1834	0.0831	0.5438	0.0169	0.2731	0.3064	0.0085	0.1522	1.0296

Table 3. Beyond-accuracy results (APLT@10, ARP@10, and PL@10) for different BRSs, CFs, and HRSs on three warm-start datasets (ML-1M, Onion, and Amazon). ↑, ↓, and → next to the metric names indicate the optimal values of individual metrics for a low popularity bias are high, low, and close to zero, respectively. Bold indicates the best performing RS, † indicates significant improvement over all others (paired t-tests with $p < 0.05$ considering Bonferroni correction).

attribute this to the more extensive and informative set of item modalities compared to user modalities (see Section 4.1). As only the demographics of cold-start users are available, the difference between HRSs and CFs is less distinct. Across all three datasets and two cold-start scenarios, SiBraR outperforms all other algorithms. The difference is even more substantial when the model can leverage any number of modalities (SiBraR_{best}). This indicates the efficiency of SiBraR in leveraging multimodal information available during inference while not being dependent on interactions.

5.1.2 Beyond-accuracy. Table 3 reports the beyond-accuracy results of all algorithms on warm-start datasets. The rows again refer to the individual BRSs, CFs, and HRSs. For each dataset, we report ARP@10, APLT@10, and PL@10 test scores as columns.[16] The arrows next to the metric names indicate which values hint at a lower popularity bias of the recommendations. A lower value for ARP@10 (↓) means that less popular items are recommended. For APLT@10 (↑), a higher number refers to more tail items being recommended. PL@10 (→) measures the discrepancy between the popularity of items in users' histories and in the recommended lists. A value close to 0 indicates that RSs match users' interests of item popularity. We highlight the best result per metric in bold, and significant improvements with †.

Unsurprisingly, Rand recommends significantly more unpopular tail items than any other algorithm (lowest ARP@10; highest APLT@10) and more unpopular ones than the items in the users' histories (most negative PL@10). In contrast, Pop recommends the most popular items (highest ARP@10; lowest APLT@10), even more popular items than what users like (most positive PL@10). Besides BRSs, the extent to which algorithms exhibit biases in comparison to other algorithms depends on the dataset and metric considered: While CFs are less biased than HRSs on ML-1M, they show a greater bias toward popular items over MuBraR and SiBraR on Onion. For Amazon, all HRSs except for DropoutNet show less bias than MF, while often more than DeepMF.

---

[16]We refer the reader to the appendix for a table on Coverage@10.

(a) SiBraR_best
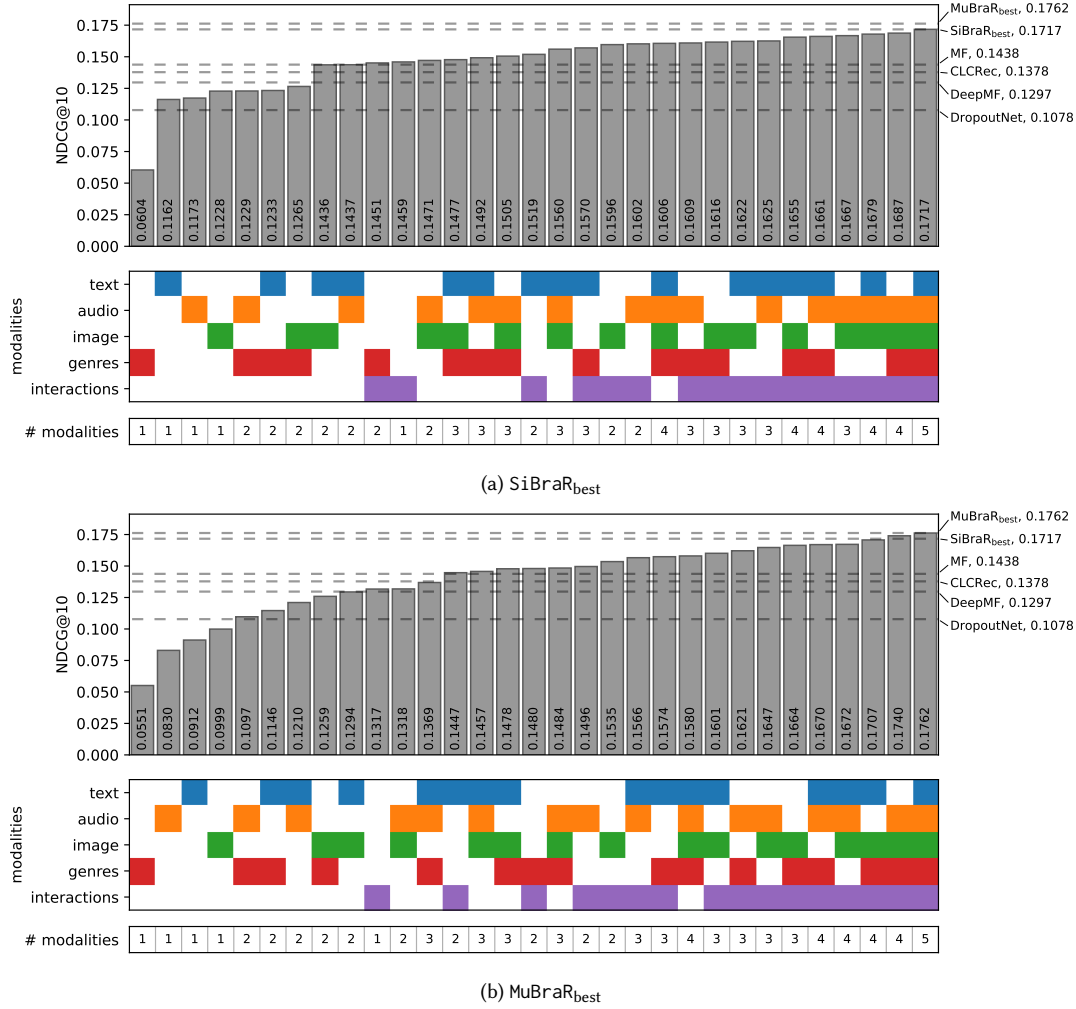
(b) MuBraR_best

Fig. 6. NDCG@10 results of (a) SiBraR_best and (b) MuBraR_best trained on warm-start Onion with interactions, genres, image, audio, and text modalities. The bars correspond to evaluations on the test set with varying evaluation modalities. If a modality is used, its block is filled with the corresponding color in the central plot. The bottom plot counts the number of modalities per bar. The gray dashed horizontal lines in the bar plot show the NDCG@10 of the best CFs and HRSs in warm-start Onion.

Comparing SiBraR and MuBraR, ARP@10 (APLT@10) is lower (higher) for both MuBraR models (MuBraR_one and MuBraR_best) on ML-1M and Amazon, while higher (lower) for Onion. Furthermore, SiBraR displays a more substantial shift toward popular items for ML-1M and Amazon, while a weaker one for Onion.

Moreover, the beyond-accuracy metrics indicate that SiBraR is one of the least popularity-biased recommenders on modality-rich datasets like Onion, and significantly less biased than others when the bias is measured by PL@10.

5.1.3 *Missing modality.* Figure 6 shows the NDCG@10 results of SiBraR_best (a) and MuBraR_best (b) models on the warm-start Onion dataset. Both models are trained on five modalities (interactions, genres, image, audio, and text) and
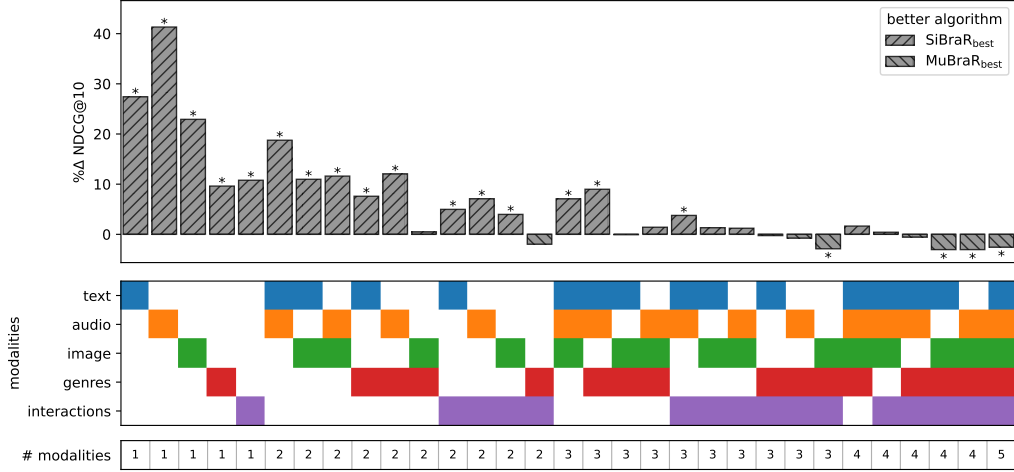
Fig. 7. NDCG@10 percent difference between SiBraR_best and MuBraR_best, both trained on warm-start Onion with interactions, genres, image, audio, and text modalities. Positive values indicate that SiBraR_best performs better than MuBraR_best on a specific set of modalities, while negative values indicate a poorer performance. An asterisk '∗' indicates a significant difference of results per subset of modalities (paired t-tests with $p < 0.05$).

evaluated on different subsets of these modalities while their weights are frozen.[17],[18] The bar plot in each subfigure presents the test scores for specific subsets of modalities. In addition, the bars are sorted by increasing score and horizontal dashed lines enable the comparison to the best warm-start CFs and HRSs. The middle plot shows which subsets of the modalities are used for evaluation. Each row corresponds to a different modality, a colored box indicates the presence of the modality during evaluation, while a missing box indicates its absence. Finally, the lower plot summarizes the number of modalities used per run.

SiBraR_$best$. Figure 6a shows the results for SiBraR_best. The plot indicates that genres are the least informative as a stand-alone modality. Moreover, text and audio are comparable in the resulting NDCG@10, slightly worse than image. By utilizing any modality except for genres, SiBraR_best achieves better results than DropoutNet. Leveraging any three modality combination or any combination containing interactions, SiBraR_best outperforms MF, DeepMF, and CLCRec. Only MuBraR_best achieves better results in warm start when all modalities are used. The utility of using interactions in recommender systems is underlined by the fact that the single-branch network only achieves peak NDCG@10 results when using interactions as part of the input. Moreover, interactions alone already lead to a stronger performance than for almost all two-pair modalities. In cold start, i.e., when no interactions are available and only the modalities {*genres, image, audio, text*} are used as input, SiBraR_best displays its strength by maintaining its advantage over MF, DeepMF, CLCRec, and DropoutNet while only suffering a slight performance decline.[19]

MuBraR_$best$. Figure 6b shows the results for MuBraR_best, for which genres again reach the lowest NDCG@10 when used alone, followed by audio, text, and image. Using interactions or any combination of two modalities, MuBraR_best achieves a higher NDCG@10 than DropoutNet. When any set of at least three modalities (except for {*genres, audio,*

---

[17]We choose the Onion dataset for a fine-grained analysis considering its richer set of modalities compared to the other datasets.
[18]We refer the reader to the appendix for plots on the other variants and datasets.
[19]This is observable in Figure 6 by considering the column for which all but the block for interaction are colored.

| | Warm-Start | | | Item Cold-Start | | | User Cold-Start | |
	ML-1M	Onion	Amazon	ML-1M	Onion	Amazon	ML-1M	Onion
MuBraR	**0.2610**	0.1259	0.0459	0.2349	0.2041	0.1225	0.4579	0.0886
MuBraR$_S$	0.2427	0.1586	0.0508	0.2620	0.2257	0.1078	0.4549	0.0902
MuBraR$_{SC}$	0.2603	**0.1762**	0.0535	0.2949	0.2276	0.1262	0.4501	0.0951
SiBraR$_S$	0.2568	0.1550	0.0615	**0.3046**†	0.2195	0.0921	0.4639	**0.1125**
SiBraR$_{SC}$	0.2581	0.1717	**0.0728**†	0.2994	**0.2297**	**0.1479**†	**0.4659**	0.1094

Table 4. NDCG@10 results for the best-performing SiBraR and MuBraR variants on different warm- and user and item cold-start datasets (ML-1M, Onion, and Amazon). Bold indicates the best-performing RSs, † indicates significant improvement over all other variants (paired t-tests with $p < 0.05$ and Bonferroni correction). The results of the different data splits are not comparable due to the different numbers of users and items in the test splits.

text}) is available, or for interactions and any additional modality, MuBraR$_{best}$ outperforms DeepMF, CLCRec, and MF. Finally, as long as only text is missing, MuBraR$_{best}$ performs better than warm-start SiBraR$_{best}$.

Comparison. To better understand differences between single- and multi-branch networks, we directly compare the NDCG@10 results of SiBraR$_{best}$ and MuBraR$_{best}$ for specific sets of modalities. Figure 7 shows the percentage difference in NDCG@10 (%ΔNDCG@10) between both algorithms, where we sort the bars by the number of modalities used at test time. A positive and negative value indicate that SiBraR$_{best}$ performs better and worse than MuBraR$_{best}$, respectively. Furthermore, an asterisk ∗ indicates a statistically significant difference of results. Especially when only small subsets of all modalities are available, the single-branch network shows its merits by achieving significant performance gains in comparison to its multi-branch counterpart, which is the case in 16 out of 31 modality subsets. In detail, the NDCG@10 scores improve by up to 40%. When more modalities become available, the difference in recommendation quality decreases, and the models perform comparably. Finally, MuBraR$_{best}$ is significantly better than SiBraR$_{best}$ in three cases, in each of which both algorithms utilize {*interactions, genres, image*} in addition to one or more other modalities as input, e. g., when all modalities are available.

Takeaway on RQ1. To investigate how SiBraR compares to other algorithms on recommendation performance (RQ1), we studied the results on different data scenarios. Our evaluation across accuracy and beyond-accuracy metrics revealed that SiBraR exhibits comparable performance to other HRSs and specifically to MuBraR in warm- and cold-start scenarios. Notably, single-branch recommenders have an advantage over multi-branch recommenders in settings with missing modalities in which SiBraR outperforms its competitors. Overall, the findings demonstrate the effectiveness of single-branch recommenders in leveraging any available data, which we attribute to the effectiveness of weight-sharing in combining multimodal data [69]. Moreover, as modality sampling serves as a way to model missing modality during training, it may improving cold-start performances.

5.2 Training paradigms

In this section, we aim to gain insights on modality sampling and contrastive loss by analyzing their effects on single- and multi-branch architectures, therefore addressing RQ2.

	ML-1M			Onion			Amazon		
	ARP↓	APLT↑	PL→	ARP↓	APLT↑	PL→	ARP↓	APLT↑	PL→
MuBraR	**0.1701**†	0.1017	**0.4240**†	**0.0142**†	**0.3520**†	**0.0999**†	0.0065	**0.3061**†	0.5160
MuBraR$_S$	0.1840	0.0659	0.5523	0.0193	0.2131	0.5138	**0.0065**	0.2459	**0.4874**†
MuBraR$_{SC}$	0.1742	0.0926	0.4584	0.0203	0.1871	0.5870	0.0081	0.2159	0.8433
SiBraR$_S$	0.1718	**0.1175**†	0.4434	0.0179	0.2390	0.4084	0.0087	0.1218	1.0685
SiBraR$_{SC}$	0.1834	0.0831	0.5438	0.0169	0.2731	0.3064	0.0085	0.1522	1.0296

Table 5. Beyond-accuracy results (APLT@10, ARP@10, and PL@10) for different MuBraR and SiBraR variants on three warm-start datasets (ML-1M, Onion, and Amazon). ↑, ↓, and → next to the metrics indicate the optimal values of individual metrics for a low popularity bias are high, low, and close to zero, respectively. Bold indicates the best performing RS, † indicates significant improvement over all others (paired t-tests with $p < 0.05$ considering Bonferroni correction).

5.2.1 Accuracy. Table 4 shows the impact of the sampling (S) and the contrastive loss (C) on single- and multi-branch networks in terms of NDCG@10.[20] SiBraR$_{SC}$ and SiBraR$_S$ describe a single-branch network with and without contrastive loss, respectively. Both versions are outlined in Figure 1. Further, MuBraR (see Figure 2) corresponds to a vanilla multi-branch network. Finally, MuBraR$_S$ features additional modality sampling and MuBraR$_{SC}$ features both, modality sampling and a contrastive loss (see Figure 3). We refer the reader to Section 3 for more detail on the algorithms. In line with Table 2, we evaluate all variants on the warm- and cold-start datasets, select the runs performing best on the validation set, and report their NDCG@10 results on the test set in Table 4. In warm start, the best-performing variant strongly depends on the dataset: For ML-1M, MuBraR has a slight advantage over MuBraR$_{SC}$ and SiBraR$_{SC}$. For Onion, MuBraR$_{SC}$ is the best-performing model, followed by SiBraR$_{SC}$ and MuBraR$_S$. Finally, SiBraR$_{SC}$ is significantly better than the other variants for Amazon, followed by SiBraR$_S$ and MuBraR$_{SC}$. The strength of SiBraR becomes apparent in user and item cold-start scenarios. Here, for each recommendation domain (movie, music, products), SiBraR$_{SC}$ outperforms the MuBraR variants. Comparing different variants in each model class, employing modality sampling in MuBraR increases its NDCG@10 five out of eight times across datasets and scenarios. Additionally, using a contrastive loss further improves the results in seven out of eight cases (all except for user cold start on ML-1M), featuring a performance improvement of MuBraR$_{SC}$ over MuBraR six out of eight times. Similarly, for single-branch networks, which use modality sampling by default, including a contrastive loss increases NDCG@10 scores of SiBraR$_{SC}$ over SiBraR$_S$ six out of eight times. Overall, the results indicate that modality sampling and a contrastive loss can strongly improve the accuracy of single- and multi-branch architectures across all datasets, for warm and cold start alike.

5.2.2 Beyond-accuracy. Analogously to Section 5.1.2 and Table 3, we evaluate different variants of single- and multi-branch networks on all warm-start datasets for popularity bias. Table 5 reports the results for ARP@10, APLT@10 and PL@10. Studying different multi-branch variants, MuBraR demonstrates the lowest popularity biases for all metrics on ML-1M and Onion. For Amazon, it shows comparable results to MuBraR$_S$ on ARP@10 and PL@10, while better results on APLT@10. On the other hand, MuBraR$_{SC}$ exhibits a lower popularity bias than MuBraR$_S$ on ML-1M, while a higher one on Onion and Amazon. In contrast, SiBraR$_{SC}$ is more biased than SiBraR$_S$ on ML-1M, while less biased on Onion and Amazon. Due to the disparity between popularity bias and recommendation quality, we leave it up to practitioners to trade-off which of the variants to employ in practice.

---

[20]We again refer the reader to the appendix for tables on all other accuracy metrics.

(a) MuBraR$_{SC}$ vs. MuBraR

(b) SiBraR$_{SC}$ vs. SiBraR$_S$
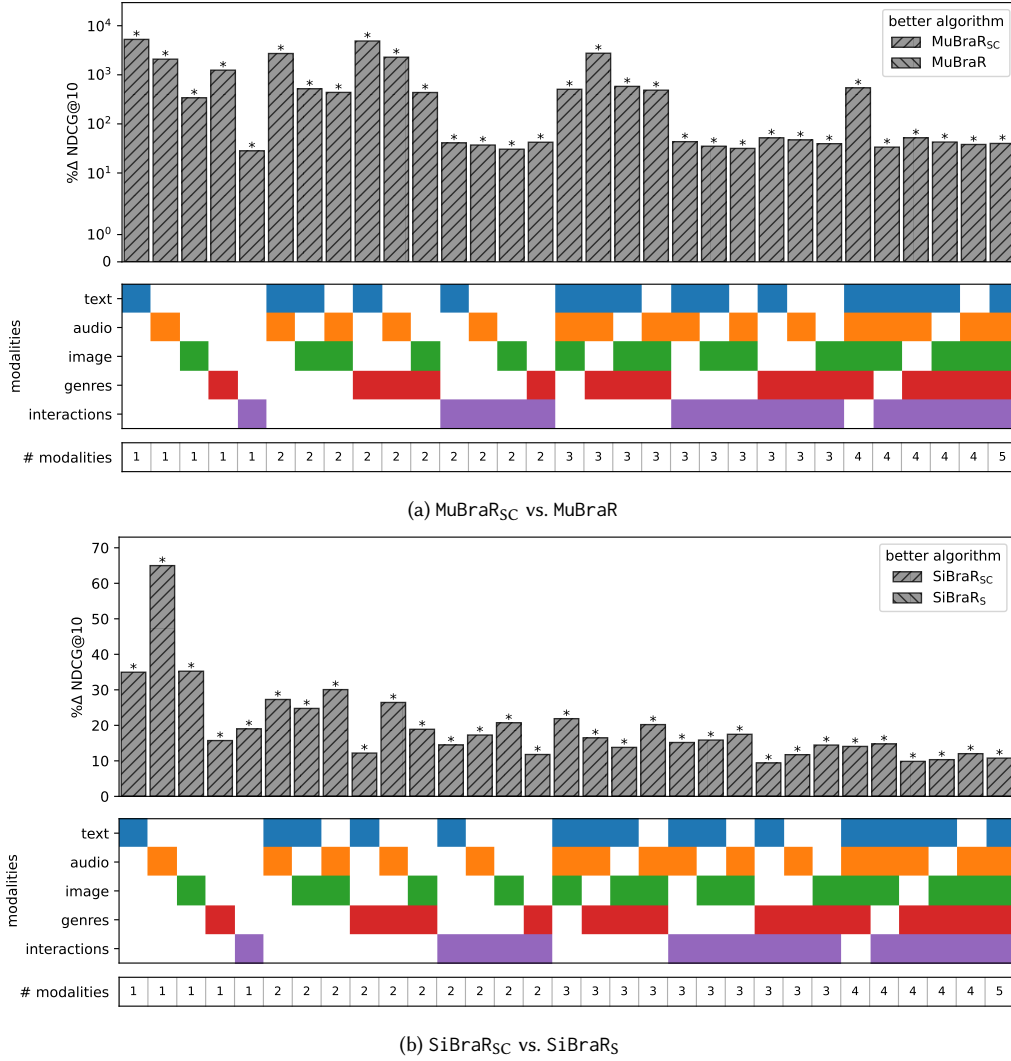
Fig. 8. NDCG@10 comparison for single- and multi-branch architectures when trained on all modalities of warm-start Onion, with and without utilizing modality sampling (S) and a contrastive loss (C). (a) shows the difference between MuBraR and MuBraR$_{SC}$, and (b) the difference between SiBraR$_S$ and SiBraR$_{SC}$. The colored squares in the middle plot indicate the modalities used during evaluation. Moreover, the bar plot shows the percent difference of NDCG@10, where positive values indicate that a contrastive loss is beneficial for recommendation quality, and worse if negative. We use a logarithmic scale for the y-axis of (a) to ease interpretation.

5.2.3 Missing modality. Figure 8 compares the percentage difference in NDCG@10 between the best single- and multi-branch networks trained on five modalities (interactions, genres, image, audio, and text) with and without modality sampling (S) and contrastive loss (C) on warm-start Onion.

MuBraR. Specifically, Figure 8a shows the difference between MuBraR$_{SC}$ and MuBraR. Here, we sort the bars by the number of modalities used at test time. We use a logarithmic scale for the y-axis to accommodate the large performance disparity between MuBraR and MuBraR$_{SC}$ in missing modality. Considering the significant improvements for all modality
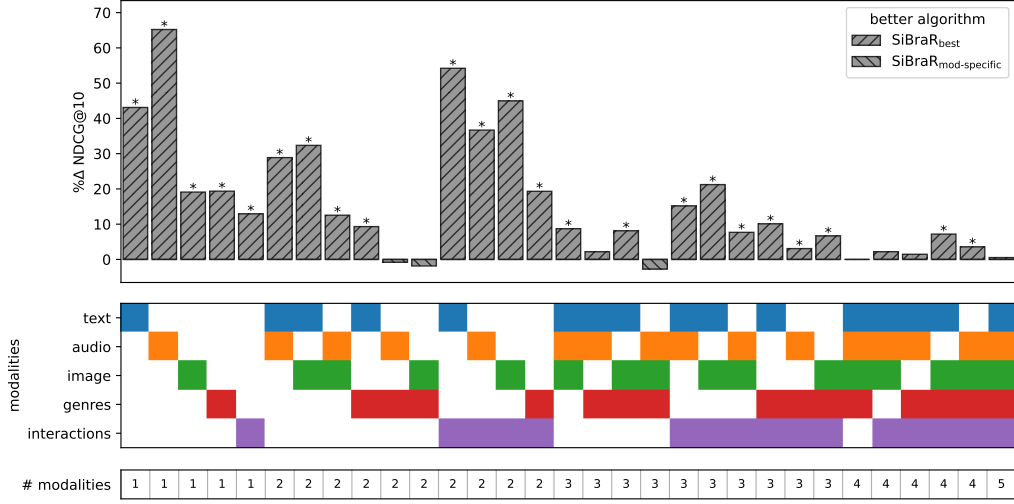
Fig. 9. NDCG@10 comparison between SiBraR_best trained on all modalities of warm-start Onion, and multiple instances of the same model (and hyperparameters) trained on specific subsets of modalities (SiBraR_mod-specific). The modalities are indicated by the colored squares in the middle plot. The bar plot shows the percent difference of NDCG, where positive values indicate that SiBraR_best performs better than SiBraR_mod-specific on the given set of modalities and worse if negative.

	# item modalities				
	1	2	3	4	5
MuBraR	0.0264	0.0504	0.0751	0.0995	0.1239
MuBraR$_S$	0.0572	0.0999	0.1272	0.1457	0.1586
MuBraR$_{SC}$	0.0921	0.1328	0.1546	**0.1674**	**0.1762**
SiBraR$_S$	0.0846	0.1186	0.1363	0.1478	0.1550
SiBraR$_{SC}$	**0.1124**	**0.1424**	**0.1574**	0.1658	0.1717
SiBraR$_{mod-specific}$	0.0876	0.1187	0.1450	0.1601	0.1697

Table 6. NDCG@10 results of different SiBraR and MuBraR variants trained on warm-start Onion with the same set of item modalities (interactions, genres, image, audio, and text) and evaluated on subsets of those. NDCG@10 values of different subsets with the same number of modalities are averaged. Bold indicates the best-performing algorithm per the number of modalities used.

	# item modalities			
	1	2	3	4
MuBraR	0.0748	0.1247	0.1674	0.2041
MuBraR$_S$	0.1069	0.1620	0.1996	0.2257
MuBraR$_{SC}$	0.1194	0.1676	0.2023	0.2276
SiBraR$_S$	0.1217	0.1640	0.1959	0.2195
SiBraR$_{SC}$	**0.1338**	**0.1762**	**0.2078**	**0.2297**

Table 7. NDCG@10 results of different SiBraR and MuBraR variants trained on cold-start Onion with the same set of item modalities (interactions, genres, image, audio, and text) and evaluated on subsets of those (excluding interactions). NDCG@10 values of different subsets with the same number of modalities are averaged. Bold indicates the best-performing algorithm per the number of modalities used.

combinations, the results indicate a strong positive influence of both sampling and contrastive loss on the multi-branch network. Moreover, the NDCG@10 improvement tends to be more pronounced when many modalities are unavailable (left side of the figure).

SiBraR. Analogously, Figure 8b summarizes the differences in performance for SiBraR. Again, utilizing a contrastive loss leads to significant improvements in NDCG. Its benefits are especially useful when modalities are scarce. Overall, the results indicate a positive influence of sampling and contrastive loss on single- and multi-branch recommenders.

Training in missing modality. Next, considering that only a subset of the modalities may be available at inference, we compare the recommendation quality of SiBraR trained on all available modalities to instances trained only on those modalities that are also available during inference. Note that training on a smaller set reduces the computational resources required for training and is thus an important aspect for practitioners. Hence, we compare $SiBraR_{best}$, the best single-branch network trained on five modalities of warm-start Onion (see Section 5.1.1) to newly initialized SiBraR instances trained on specific subsets of modalities, while using the same hyperparameters as $SiBraR_{best}$. We collectively refer to these networks as $SiBraR_{mod-specific}$. Figure 9 reports the percentage change in NDCG@10 of $SiBraR_{best}$ over $SiBraR_{mod-specific}$ per subset of modalities. While the performance in modality-rich scenarios is comparable[21], SiBraR performs significantly better than specific $SiBraR_{mod-specific}$ instances across almost all modalities subsets. This result illustrates that training single-branch networks on all available modalities may be beneficial for inference, even if many modalities are unavailable and the training is more resource intensive.

Conclusion on missing modality. Table 6 summarizes all previous observations by reporting the average NDCG@10 scores of single- and multi-branch variants on warm-start Onion of MuBraR, all trained on five modalities and evaluated on all possible subsets of these modalities. While each row represents a different variant, each column corresponds to the number of modalities used during evaluation. The reported values are an average of the results over all possible combinations of a specific number of modalities. The table shows the merits of modality sampling, contrastive loss, and of the single-branch architecture. Across all numbers of modalities, $MuBraR_{SC}$ is, on average, better than $MuBraR_S$, which in turn is, on average, better than MuBraR. Similarly, $SiBraR_{SC}$ is better than $SiBraR_S$. Moreover, $SiBraR_{SC}$ is, on average, substantially better than $MuBraR_{SC}$ when only few modalities are available and comparable when at most two modalities are missing. Finally, $SiBraR_{mod-specific}$ only achieves similar results to $SiBraR_{SC}$ in warm-start. Conducting a similar analysis on item cold-start Onion, Table 7 indicates similar results: $SiBraR_{SC}$ shows a solid improvement for low modality counts, and this improvement vanishes with an increasing number of modalities. We refer the reader to the appendix for similar analyses on ML-1M and Amazon.

5.2.4 Contrastive loss. This section aims to understand the impact and effect of the contrastive loss as well as its two parameters on SiBraR: (1), the weight α, which balances recommendation and contrastive loss terms (Equation 5), and (2), the temperature τ controlling the parametrization of the $\mathcal{L}_{SInfoNCE}$ loss (Equation 3).

To this end, we perform a grid search over various combinations of contrastive weight α and temperature τ on warm-start Onion, selecting the hyperparameters of $SiBraR_{best}$ as the basis for this experiment. Specifically, fixing all other hyperparameters, we only vary α and τ, train a new model from scratch, and assess its accuracy and modality gap on the test set.

---

[21]The slight discrepancy is due to randomness in negative and modality sampling.
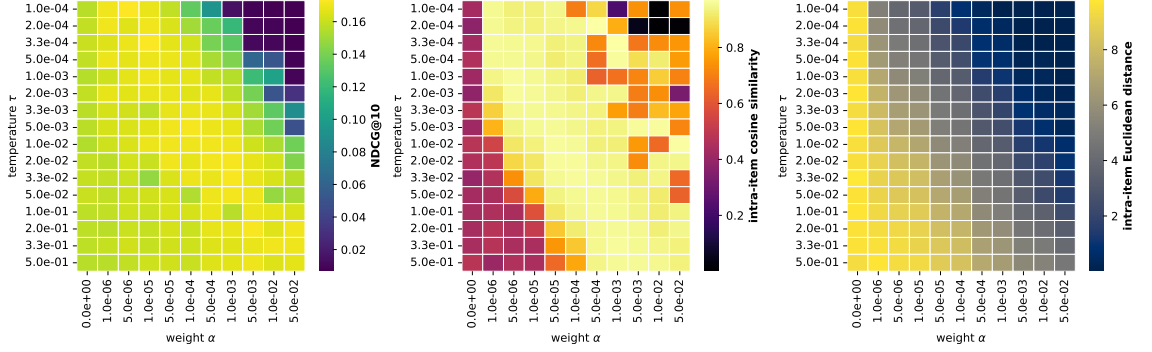
Fig. 10. NDCG@10, intra-item cosine similarity and Euclidean distance results of SiBraR$_{best}$ on different combinations of contrastive loss temperature τ and weight α on warm-start Onion, while all other hyperparameters are fixed. The left plot shows the NDCG@10 results at test time, and the center and right plots show the intra-item consine similarity and Euclidean distance, respectively. When contrastive loss $\alpha = 0$, i. e., when the loss does not affect training, the minor performance differences are due to randomness in the training pipeline.
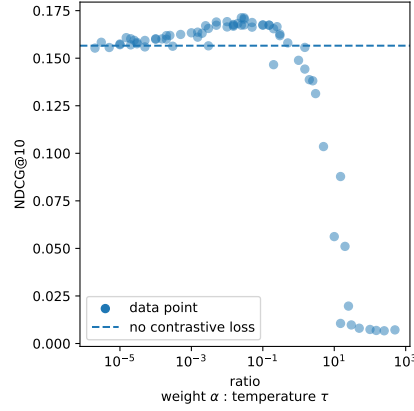
Fig. 11. NDCG@10 results of SiBraR$_{best}$ on different combinations of contrastive loss temperature τ and weight α on warm-start Onion, while all other hyperparameters are fixed. Each point indicates the performance of a run with a certain ratio α/τ and the resulting NDCG@10. The horizontal dashed line shows the average performance for runs where the contrastive loss has no effect on training, i. e., when $\alpha = 0$.

Figure 10 reports the NDCG@10, intra-item cosine similarity and intra-item Euclidean distance results (see Section 4.4.3) of different weight–temperature combinations as heatmaps. The left plot correspond to NDCG@10, and the middle and right plot to the intra-item cosine similarity and Euclidean distance, respectively. Moreover, the weight α and temperature τ are plotted logarithmically on the x- and y-axis in each heatmap. The colored bar to the right of each plot outlines the value ranges of the resulting metrics.

NDCG@10. Considering the NDCG@10 (left-most plot in Figure 10), a subregion of well-performing experiments spans diagonally across the heatmap, from low temperatures and low weights to high temperatures and high weights. In contrast, experiments with very low weights and high temperatures achieve suboptimal results, similar to ones

with high weights and low temperatures. To further understand the relationship between temperature and weight, Figure 11 reports the ratio α/τ of the tested weights and temperatures on the X-axis and the NDCG@10 result of each run n the y-axis. The horizontal dashed line corresponds to the average performance of models with weight $\alpha = 0$ (i. e., no contrastive loss). The results indicate the presence of an optimal proportion for the contrastive loss parameters to improve the performance of single-branch networks – for Onion between 10^{-3} and 10^{-1}.[22] Using this insight for training, fixing a reasonable value for either weight or temperature and scanning over a broad range for the other parameter might simplify the hyperparameter search.

Intra-Item cosine similarity. Analogous to the previous paragraph, the middle plot in Figure 10 shows the intra-item cosine similarities on warm-start Onion. We again observe a diagonal-spanning subregion with very high similarity values. Off-diagonal combinations, or when the contrastive loss has no influence ($\alpha = 0$), exhibit the lowest similarity values. Finally, the diagonal high-similarity subregion roughly aligns with the previous one on NDCG@10, hinting at a positive relationship between intra-item modality similarity and model accuracy.

Intra-Item Euclidean distance. The right plot of Figure 10 demonstrates the direct effect of contrastive weight and temperature on the shared embedding space of $\texttt{SiBraR}_{\text{SC}}$. The plot indicates that with decreasing temperature and increasing weight, different modalities of the same items are further pushed together.

Takeaway on RQ2. In the above, we studied the effects of the different training paradigms employed in $\texttt{SiBraR}$ (RQ2). The results suggest that while modality sampling alone may be harmful for model performance in warm start, its merits are evident in missing modality scenarios. This is to be expected, since modality-sampling can be considered as a way to model missing modality during training. Employing a contrastive loss alongside the modality sampling benefits single- and multi-branch networks alike; both exhibit significantly better results across warm-start and missing-modality scenarios. The improved performance can be explained by the fact that the contrastive loss enhances the similarity between the representations of each modality and that of the collaborative signal, rendering each of them more effective for recommendation. Finally, the optimal choice of the contrastive loss parameters (i. e., weight and temperature) is essential for training $\texttt{SiBraR}$. While combinations of low weights and high temperatures may lead to suboptimal performances in terms of accuracy, high weights might harm the recommendation objective of $\texttt{SiBraR}$. The potential negative impact of the contrastive loss on recommendation is to be attributed to the fact that both high weights and low temperatures lead to larger components of the contrastive loss in the overall loss function; these high values could therefore render the signals from the recommendation loss negligible during training.

5.3 Modality gap

In this section, we investigate the ability of single- and multi-branch networks to reduce the modality gap by mapping different modalities of the same items into similar regions in the embedding space.[23] This corresponds to answering RQ3.

5.3.1 Qualitative analysis. For our analysis, we consider single- and multi-branch variants trained on five item modalities (interactions, genres, audio, image, and text) of warm-start Onion. Figure 12 shows the input (left) and output (right) embedding space of the single-branch network g of $\texttt{SiBraR}_{\text{SC}}$. To visualize the higher-dimensional embeddings in 2D,

---

[22]This proportion likely also depends on the model class and task-specific losses, which the contrastive loss has to be weighed against.
[23]We note that studying the modality gap is limited to multimodal algorithms with shared embedding spaces. Thus, as DropoutNet concatenates it's modalities, we do not study it. Moreover, as our main interest lies in understanding the effects of weight sharing, modality sampling and contrastive loss, we refer the reader to the appendix for results on CLCRec.
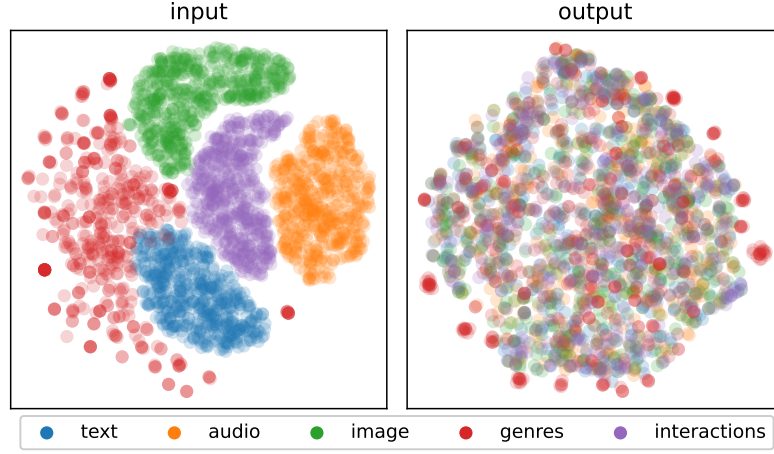
Fig. 12. T-SNE projected embeddings before and after the single-branch network of the best performing SiBraR_SC on warm-start Onion. While different modalities can be differentiated at input (left), modalities overlap substantially in the shared embedding space after passing the modalities through the network (right).
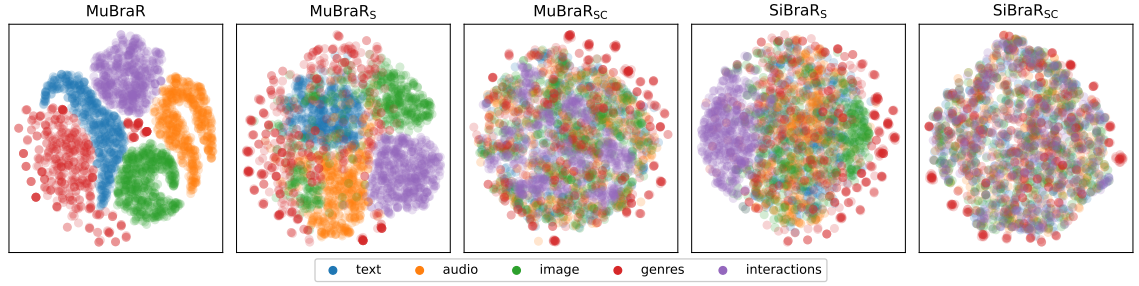
Fig. 13. T-SNE projected embeddings of the modality-specific embeddings of the best single- and multi-branch variants trained on warm-start Onion. While the different modalities can easily be differentiated for MuBraR (1st), due to modality sampling (MuBraR_S, 2nd) and the contrastive loss (MuBraR_SC, 3rd), the different modalities are less separable. For SiBraR_S (4th from left), the regions of the different modalities overlap, though not as much as for SiBraR_SC (rightmost).

we consider only the first 10 principal components obtained through principal component analysis (PCA). We do so to reduce the computational cost for t-Distributed Stochastic Neighbor Embedding (t-SNE) [21], which we apply afterward to further reduce the dimensionality to 2. For PCA and t-SNE, we use the implementations and default parameters provided by the *scikit-learn* [8] library.[24] To enhance perceptibility, we only show the projections of a subset of all items by sampling 500 items uniformly at random.[25]

The left plot in Figure 12 shows that before applying the single-branch network, all modalities cover separate regions in the input space. In contrast, in the output of the single-branch network (right), the individual regions overlap. The worsened separability indicates that the embeddings of different modalities of SiBraR_SC become more similar, which, in turn, the downstream recommendation task can profit from.

---

[24]https://scikit-learn.org/
[25]See Figure 21 in the appendix for the projections of ML-1M and Amazon.

	ML-1M		Onion		Amazon	
	intra	inter	intra	inter	intra	inter
MuBraR	0.136	0.489	0.032	0.486	0.063	**0.772**
MuBraR$_S$	0.308	0.333	0.288	0.282	0.707	0.423
MuBraR$_{SC}$	0.456	0.272	0.498	0.329	**0.899**	0.405
SiBraR$_S$	0.718	**0.668**	0.478	0.353	0.715	0.327
SiBraR$_{SC}$	**0.845**	0.615	**0.929**	**0.854**	0.895	0.323

Table 8. Average inter- and intra-item cosine similarity for SiBraR and MuBraR variants. For each warm-start dataset, all variants are trained and evaluated on the same set of modalities: (interactions, genres, text) for ML-1M, (interactions, genres, audio, image, text) for Onion, and (interactions, text) for Amazon. Bold indicates the algorithm where modality embeddings exhibit the highest average similarity within each group.

Similarly, Figure 13 shows projections of the embedding spaces of all single- and multi-branch variants. While different modalities embedded by MuBraR form individual subspaces in their joint embedding space, these spaces start overlapping for MuBraR$_S$, which indicates a positive influence of modality sampling to reduce the modality gap in multi-branch networks. Finally, the projections of MuBraR$_{SC}$ suggest that employing a contrastive loss benefits MuBraR by encouraging its independent branches to map different modalities to the same region in the embedding space. This observation is further supported by comparing the projections of SiBraR$_S$ and SiBraR$_{SC}$. In SiBraR$_{SC}$, the modalities exhibit reduced separability in the embedding space, indicating more similar representations.

5.3.2 Quantitative analysis. To verify the above observations, we quantitatively evaluate the individual embedding spaces of the different SiBraR and MuBraR variants on all warm-start datasets.

Cosine similarity. Table 8 reports the average intra- and inter-item modality cosine similarities (see Section 4.4.3).[26] Each row corresponds to the best model of a specific variant. The columns indicate the dataset and whether they have been computed between different modalities of the same item (intra) or between the same modalities of different items (inter). The cosine similarity is bound by −1 and 1, where values close to -1 and 1 indicate high and low similarity, respectively. We indicate the algorithm with the highest average similarity between modalities in bold. Finally, the reported values are an average over the measurements between all possible modality pairs for the respective metrics. As the interpretation for a single result is difficult, we focus on comparing *intra* and *inter* results to draw insights.

For MuBraR, inter-item cosine similarities are higher than intra-item similarities for all datasets. For MuBraR$_S$, which includes sampling, the iter-item cosine similarity is lower than the intra-item similarity for ML-1M, while lower for Onion and Amazon. For MuBraR$_{SC}$, which is further equipped with a contrastive loss, and for single-branch networks SiBraR$_S$ and SiBraR$_{SC}$, the intra-item cosine similarities are consistently lower than the inter-item similarity for all datasets. Moreover, the intra-item values continuously increase when employing sampling and a contrastive loss alongside MuBraR and SiBraR. Except for the SiBraR models on Onion, an increase in intra-item cosine similarity coincides with a decrease in inter-item cosine similarity. Finally, for ML-1M and Onion, SiBraR$_{SC}$ achieves a substantially higher intra-item similarity than all other variants and a comparable similarity for Amazon. Our results indicate that by utilizing modality sampling and a contrastive loss, the single-branch forces multimodal models to push modality representations of the same items closer together than representations of different items.

---

[26]We refer the reader to the appendix for results on Euclidean distance.

	ML-1M	Onion	Amazon
MuBraR	1.0000	1.0000	1.0000
MuBraR$_S$	1.0000	1.0000	1.0000
MuBraR$_{SC}$	1.0000	0.9925	0.8853
SiBraR$_S$	0.9768	0.9298	0.9811
SiBraR$_{SC}$	**0.8983†**	**0.7771†**	**0.6823†**
Random	0.3333	0.2000	0.5000

Table 9. Accuracy of random forest classifiers predicting the modality class of modality embeddings output by MuBraR and SiBraR variants trained on three warm-start datasets (ML-1M, Onion, and Amazon). Each result reports the average performance of 20 classifiers trained on different train–test splits and seeds. 80% of each warm-start test set is used for fitting the classifiers, and 20% for their evaluation. Bold indicates the variant with the least separable embedding space, † indicates significant worse classifier accuracies in comparison to all other methods (paired t-tests with $p < 0.05$ considering Bonferroni correction). The last row shows the expected accuracy for random guessing.

Modality prediction. We now study whether the aforementioned shared embedding spaces express the interchangeability of different modalities. To this purpose, we study whether it is possible to determine to which modality type a given embedding vector belongs to. Table 9 reports the average accuracy of random forest classifiers[27] trained on this classification task. Each row in the table corresponds to a different single- and multi-branch variant. Moreover, each column reports the results for a specific warm-start dataset. We use the embeddings of the datasets' test splits and further split them for our classification experiment into train and test items with ratios 80:20 by sampling them uniformly at random. Next, we fit a random forest classifier on each train set and evaluate it on the test set. For stronger validity of the results, we repeat this process of dataset splitting and classifier training with 20 different seeds and report the average accuracy of all classifiers. We also report the expected accuracy of a random predictor. Bold values indicate the the algorithms with the least separable embeddings, while † that the classifiers perform significantly worse than on other methods. For MuBraR and MuBraR$_S$, the classifiers are able to perfectly discern the modality types of different embeddings (accuracy = 1), indicating low similarity between the embeddings. The classification task is slightly more difficult for MuBraR$_{SC}$, for which classifiers are not able to predict the modalities perfectly on Onion and Amazon. Finally, the results of SiBraR$_S$ and SiBraR$_{SC}$ highlight similarities in the modality embeddings in the embedding space, making discerning the modality classes more challenging. Although the classifiers trained on SiBraR$_{SC}$ embeddings exhibit statistically significant lower accuracies, their performance remains substantially above random. This suggests persistent semantic differences between modality types, and together with previous insights from Section 5.1.3 and Section 5.2.3, the benefits of utilizing all available modalities for generating recommendations.

Takeaway on RQ3. In the section, we investigated the ability of single- and multi-branch networks and their training paradigms to reduce the modality gap between different modalities of the same items (RQ3). Our qualitative and quantitative analysis indicates that modality sampling, contrastive loss, and the single-branch network have positive effects on reducing the modality gap by encouraging the models to encode different modalities of the same item more closely together while modalities of different items further apart. Moreover, applying these paradigms increases the difficulty in determining the modality for a given embedding, further suggesting an increase in modality similarity.

---

[27]We use the implementation of scikit-learn [8] with default parameters (https://scikit-learn.org/1.5/modules/generated/sklearn.ensemble.RandomForestClassifier.html).

Overall, the results suggest that reducing the modality gap is a potential factor for increased model performance in missing modality scenarios.

6 Conclusion

This work formalizes multimodal single- and multi-branch recommenders, `SiBraR` and `MuBraR`, respectively. With extensive experiments, we quantitatively compare these approaches on three datasets on warm-start, missing modality, and user and item cold-start scenarios. Besides accuracy metrics, we investigate different popularity biases these models exhibit. We break down the factors contributing to strong recommendations employed in `SiBraR`: weight-sharing, modality sampling, and contrastive loss. As the results show, single- and multi-branch recommenders perform comparably well in warm-start, while the former have a clear advantage in cold start and missing modality with an, on average, more than 10% increase in NDCG@10 for low numbers of available modalities. Utilizing a contrastive loss is paramount for achieving strong results, even when parts of the training data are missing at inference. We perform a widespread analysis on contrastive loss parameters to shed additional light on their importance. The study reveals that the right choice of parameters may considerably improve the recommendation performance of a model. Finally, we qualitatively and quantitatively examine the modality gap of embedding spaces of single- and multi-branch approaches. Specifically, we show that modality sampling and contrastive loss affect the distance between embeddings of the same item and of different items; both force modality embeddings of the same items to be closer to each other than to embeddings of other items. Classification, recommendation and other downstream tasks utilizing the embeddings may thus be less affected by the underlying modality source.

6.1 Practical considerations

As outlined in this work, single- and multi-branch networks allow for generating recommendations by utilizing any available combination of modalities used during training. While results in missing-modality scenarios are worse than in warm start, we show that single-branch networks significantly outperform their competitors in the former case. Moreover, users and items with missing modalities are often dropped from the datasets before training. In such cases, the presented modality sampling and contrastive loss naturally allow training on missing-modality data. To reduce computational resources at inference, `SiBraR` enables practitioners to precompute embeddings of static modalities, e. g., audio or lyrics in the music domain, while (re)computing the embeddings for previously unavailable or updated modalities, e. g., interactions. Furthermore, practitioners can trade off recommendation quality and computational costs by selecting only a subset of all modalities for recommendation and utilizing left-out modalities only to ensure a minimum recommendation quality.

6.2 Limitations and Future Directions

The use of single- and multi-branch losses for recommendation poses several challenges and limitations. Compared to other models, both single- and multi-branch recommenders come at the cost of a large number of hyperparameters. We experienced that these models are prone to substantial drops in recommendation quality when the hyperparameters are not optimized; hence tuning single- and multi-branch architectures might require large computational costs. We therefore believe that our analysis on the impact of the contrastive-loss hyperparameters will be helpful in simplifying the tuning phase in future applications. There are still many open research questions pertaining to single-branch recommenders, which are too extensive for this work to cover. For example, we only sample pairs of modalities when applying the constrastive loss; the effect of sampling multiple pairs at once is not studied. We only considered random

sampling of multimodal data for all items during training; this might not be reflective of real-world scenarios in which not all modalities are equally available for all items. Further, we only considered averaging as a modality fusion strategy for both single- and multi-branch networks, despite many different techniques, e. g., attention-based fusion, emerging in the domain of multimodal learning. Moreover, while `SiBraR` and `MuBraR` models may feature similar numbers of parameters, the former require multiple forward passes – one for each modality – through the single branch. On the other hand, the latter require only forward passes through modality-specific networks, which are comparably smaller in size. The dynamics of including additional modality types on a trained `SiBraR` instance remain unknown. We limited our study to the beyond-accuracy evaluation in warm-start scenarios where all modalities are available. An additional in-depth analysis on the introduced biases of different modalities may provide insights on how cold-start users and items are inherently affected by the modalities used to represent them in the absence of collaborative information. We envision studying the practical implications and addressing the limitations as future extensions of this work.

Acknowledgments

References

[1] Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. 2019. Managing Popularity Bias in Recommender Systems with Personalized Re-Ranking. In *Proceedings of the Thirty-Second International Florida Artificial Intelligence Research Society Conference* (Florida, USA). AAAI Press, 413–418. https://aaai.org/ocs/index.php/FLAIRS/FLAIRS19/paper/view/18199

[2] Himan Abdollahpouri, Masoud Mansoury, Robin Burke, and Bamshad Mobasher. 2020. The Connection Between Popularity Bias, Calibration, and Fairness in Recommendation. In *Proceedings of the 14th ACM Conference on Recommender Systems (RecSys)* (Virtual Event, Brazil). Association for Computing Machinery, 726–731. doi:10.1145/3383313.3418487

[3] Omer Arshad, Ignazio Gallo, Shah Nawaz, and Alessandro Calefati. 2019. Aiding intra-text representations with visual context for multimodal named entity recognition. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE, 337–342.

[4] Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. 2019. Multimodal Machine Learning: A Survey and Taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41, 2 (2019), 423–443.

[5] Oren Barkan, Noam Koenigstein, Eylon Yogev, and Ori Katz. 2019. CB2CF: a neural multiview content-to-collaborative filtering model for completely cold item recommendations. In *Proc. of ACM RecSys* (Copenhagen, Denmark). 228–236.

[6] Eric Behar, Julien Romero, Amel Bouzeghoub, and Katarzyna Wegrzyn-Wolska. 2023. Tackling Cold Start for Job Recommendation with Heterogeneous Graphs. In *Proc. of HR Workshop co-located with ACM RecSys* (Singapore, Singapore).

[7] Lukas Biewald. 2020. Experiment Tracking with Weights and Biases. https://www.wandb.com/ Software available from wandb.com.

[8] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. 2013. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning* (Prague, Czech Republic). 108–122.

[9] Desheng Cai, Shengsheng Qian, Quan Fang, Jun Hu, and Changsheng Xu. 2023. User Cold-Start Recommendation via Inductive Heterogeneous Graph Neural Network. *ACM Trans. Inf. Syst.* 41, 3, Article 64 (2023), 27 pages.

[10] Yuwei Cao, Liangwei Yang, Chen Wang, Zhiwei Liu, Hao Peng, Chenyu You, and Philip S. Yu. 2023. Multi-task Item-attribute Graph Pre-training for Strict Cold-start Item Recommendation. In *Proc. of ACM RecSys* (Singapore, Singapore). 322–333.

[11] Gaode Chen, Ruina Sun, Yuezihan Jiang, Jiangxia Cao, Qi Zhang, Jingjian Lin, Han Li, Kun Gai, and Xinghua Zhang. 2024. A Multi-modal Modeling Framework for Cold-start Short-video Recommendation. In *Proceedings of the 18th ACM Conference on Recommender Systems (RecSys)* (Bari, Italy) *(RecSys '24)*. Association for Computing Machinery, New York, NY, USA, 391–400. doi:10.1145/3640457.3688098

[12] Yashar Deldjoo, Markus Schedl, Paolo Cremonesi, and Gabriella Pasi. 2020. Recommender Systems Leveraging Multimedia Content. *ACM Comput. Surv.* 53, 5, Article 106 (sep 2020), 38 pages. doi:10.1145/3407190

[13] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. 2009. ImageNet: A Large-Scale Image Database. In *Proc. of IEEE CVPR* (Miami, FL, USA). 248–255.

[14] Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. 2020. Jukebox: A Generative Model for Music. *arXiv preprint arXiv:2005.00341* (2020).

[15] Hamid Eghbal-Zadeh, Bernhard Lehner, Markus Schedl, and Gerhard Widmer. 2015. I-Vectors for Timbre-Based Music Similarity and Music Artist Classification. In *Proc. of ISMIR* (Málaga, Spain). 554–560.

[16] Christian Ganhör, David Penz, Navid Rekabsaz, Oleg Lesota, and Markus Schedl. 2022. Unlearning Protected User Attributes in Recommendations with Adversarial Training. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Madrid, Spain) *(SIGIR '22)*. Association for Computing Machinery, New York, NY, USA, 2142–2147. doi:10.1145/3477495.3531820

[17] Christian Ganhör, Marta Moscati, Anna Hausberger, Shah Nawaz, and Markus Schedl. 2024. A Multimodal Single-Branch Embedding Network for Recommendation in Cold-Start and Missing Modality Scenarios. In *Proceedings of the 18th ACM Conference on Recommender Systems (RecSys)* (Bari, Italy). Association for Computing Machinery, New York, NY, USA, 380–390. doi:10.1145/3640457.3688138

[18] Yuqi Gong, Xichen Ding, Yehui Su, Kaiming Shen, Zhongyi Liu, and Guannan Zhang. 2023. An Unified Search and Recommendation Foundation Model for Cold-Start Scenario. In *Proc. of ACM CIKM* (Birmingham, United Kingdom). 4595–4601.

[19] F. Maxwell Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. *ACM Trans. Interact. Intell. Syst.* 5, 4, Article 19 (2015), 19 pages.

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *Proc. of IEEE CVPR* (Las Vegas, NV, USA). 770–778.

[21] Geoffrey E Hinton and Sam Roweis. 2002. Stochastic Neighbor Embedding. In *Advances in Neural Information Processing Systems*, S. Becker, S. Thrun, and K. Obermayer (Eds.), Vol. 15. MIT Press. https://proceedings.neurips.cc/paper_files/paper/2002/file/6150ccc6069bea6b5716254057a194ef-Paper.pdf

[22] Yupeng Hou, Jiacheng Li, Zhankui He, An Yan, Xiusi Chen, and Julian McAuley. 2024. Bridging Language and Items for Retrieval and Recommendation. *arXiv preprint arXiv:2403.03952* (2024).

[23] Christina Humer, Marc Streit, and Hendrik Strobelt. 2023. *Amumo (Analyze Multi-Modal Models).* https://github.com/ginihumer/Amumo

[24] Dietmar Jannach, Lukas Lerche, and Markus Zanker. 2018. *Recommending Based on Implicit Feedback.* Lecture Notes in Computer Science, Vol. 10100. Springer International Publishing, 510–569. doi:10.1007/978-3-319-90092-6_14

[25] Euiju Jeong, Xinzhe Li, Angela (Eunyoung) Kwon, Seonu Park, Qinglong Li, and Jaekyeong Kim. 2024. A Multimodal Recommender System Using Deep Learning Techniques Combining Review Texts and Images. *Applied Sciences* 14, 20 (2024). doi:10.3390/app14209206

[26] Anastasiia Klimashevskaia, Mehdi Elahi, Dietmar Jannach, Christoph Trattner, and Lars Skjærven. 2022. Mitigating Popularity Bias in Recommendation: Potential and Limits of Calibration Approaches. In *Advances in Bias and Fairness in Information Retrieval* (Stavanger, Norway). Springer International Publishing, 82–90. doi:10.1007/978-3-031-09316-6_8

[27] Anastasiia Klimashevskaia, Dietmar Jannach, Mehdi Elahi, and Christoph Trattner. 2024. A survey on popularity bias in recommender systems. *User Modeling and User-Adapted Interaction* 34, 5 (2024), 1777–1834. doi:10.1007/S11257-024-09406-0

[28] Guilin Lan, Yeqian Du, and Zhouwang Yang. 2024. Robust Multimodal Representation under Uncertain Missing Modalities. *ACM Transactions on Multimedia Computing, Communications and Applications* (2024).

[29] Yi-Lun Lee, Yi-Hsuan Tsai, Wei-Chen Chiu, and Chen-Yu Lee. 2023. Multimodal Prompting with Missing Modalities for Visual Recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 14943–14952.

[30] Guohui Li, Li Zou, Zhiying Deng, and Qi Chen. 2024. Neighborhood-Enhanced Multimodal Collaborative Filtering for Item Cold Start Recommendation. In *Proc. of ICASSP* (Seoul, Korea). 7815–7819.

[31] Victor Weixin Liang, Yuhui Zhang, Yongchan Kwon, Serena Yeung, and James Y Zou. 2022. Mind the Gap: Understanding the Modality Gap in Multi-modal Contrastive Representation Learning. In *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (Eds.), Vol. 35. Curran Associates, Inc., 17612–17625. https://proceedings.neurips.cc/paper_files/paper/2022/file/702f4db7543a7432431df588d57bc7c9-Paper-Conference.pdf

[32] Ronghao Lin and Haifeng Hu. 2023. MissModal: Increasing Robustness to Missing Modality in Multimodal Sentiment Analysis. *Transactions of the Association for Computational Linguistics* 11 (2023), 1686–1702.

[33] Qidong Liu, Jiaxi Hu, Yutian Xiao, Xiangyu Zhao, Jingtong Gao, Wanyu Wang, Qing Li, and Jiliang Tang. 2024. Multimodal recommender systems: A survey. *Comput. Surveys* 57, 2 (2024), 1–17.

[34] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in Neural Information Processing systems* 32 (2019).

[35] Mengmeng Ma, Jian Ren, Long Zhao, Davide Testuggine, and Xi Peng. 2022. Are Multimodal Transformers Robust to Missing Modality?. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 18177–18186.

[36] Mengmeng Ma, Jian Ren, Long Zhao, Sergey Tulyakov, Cathy Wu, and Xi Peng. 2021. SMIL: Multimodal learning with severely missing modality. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 2302–2310.

[37] Paul Magron and Cédric Févotte. 2022. Neural content-aware collaborative filtering for cold-start music recommendation. *Data Mining and Knowledge Discovery* 36, 5 (2022), 1971–2005.

[38] Daniele Malitesta, Emanuele Rossi, Claudio Pomo, Tommaso Di Noia, and Fragkiskos D Malliaros. 2024. Do We Really Need to Drop Items with Missing Modalities in Multimodal Recommendation?. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*. 3943–3948.

[39] Sébastien Marcel and Yann Rodriguez. 2010. Torchvision the machine-vision package of torch. In *Proc. of ACM Multimedia* (Firenze, Italy). 1485–1488.

[40] Alessandro B. Melchiorre, Navid Rekabsaz, Christian Ganhör, and Markus Schedl. 2022. ProtoMF: Prototype-based Matrix Factorization for Effective and Explainable Recommendations. In *Proc. of ACM RecSys* (Seattle, WA, USA). 246–256.

[41] Alessandro B. Melchiorre, Navid Rekabsaz, Emilia Parada-Cabaleiro, Stefan Brandl, Oleg Lesota, and Markus Schedl. 2021. Investigating gender fairness of recommendation algorithms in the music domain. *Information Processing & Management* 58, 5 (2021), 102666.

[42] Alessandro B. Melchiorre, Navid Rekabsaz, Emilia Parada-Cabaleiro, Stefan Brandl, Oleg Lesota, and Markus Schedl. 2021. Investigating Gender Fairness of Recommendation Algorithms in the Music Domain. *Information Processing & Management* 58, 5 (2021), 102666.

[43] Marta Moscati, Emilia Parada-Cabaleiro, Yashar Deldjoo, Eva Zangerle, and Markus Schedl. 2022. Music4All-Onion - A Large-Scale Multi-faceted Content-Centric Music Recommendation Dataset. In *Proc. of ACM CIKM* (Atlanta, GA, USA). 4339–4343.

[44] Peter Müllner, Elisabeth Lex, Markus Schedl, and Dominik Kowald. 2024. The Impact of Differential Privacy on Recommendation Accuracy and Popularity Bias. In *Advances in Information Retrieval* (Glasgow, United Kingdom). Springer Nature Switzerland, 466–482. doi:10.1007/978-3-031-56066-8_33

[45] Arsha Nagrani, Samuel Albanie, and Andrew Zisserman. 2018. Learnable PINs: Cross-modal embeddings for person identity. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 71–88.

[46] Arsha Nagrani, Samuel Albanie, and Andrew Zisserman. 2018. Seeing voices and hearing faces: Cross-modal biometric matching. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 8427–8436.

[47] Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying Recommendations using Distantly-Labeled Reviews and Fine-Grained Aspects. In *Proc. of EMNLP*. 188–197.

[48] Rong Pan, Yunhong Zhou, Bin Cao, Nathan N. Liu, Rajan Lukose, Martin Scholz, and Qiang Yang. 2008. One-Class Collaborative Filtering. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*. IEEE Computer Society, Pisa, Italy, 502–511. doi:10.1109/ICDM.2008.16

[49] Deepak Kumar Panda and Sanjog Ray. 2022. Approaches and algorithms to mitigate cold start problems in recommender systems: a systematic literature review. *J. Intell. Inf. Syst.* 59, 2 (oct 2022), 341–366.

[50] Claudio Pomo, Dietmar Jannach, Yubin Kim, Daniele Malitesta, Alberto Carlo Maria Mancino, Julian McAuley, Alessandro Melchiorre, and Shah Nawaz. 2025. First International Workshop on Data Quality-Aware Multimodal Recommendation (DaQuaMRec). In *Proceedings of the 19th ACM Conference on Recommender Systems (RecSys)* (Prague, Czech Republic). Association for Computing Machinery, New York, NY, USA, 1378–1382. doi:10.1145/3705328.3748001

[51] Jordi Pons, Oriol Nieto, Matthew Prockup, Erik M. Schmidt, Andreas F. Ehmann, and Xavier Serra. 2018. End-to-end learning for music audio tagging at scale. In *Proc. of ISMIR*.

[52] Jordi Pons and Xavier Serra. 2019. musicnn: pre-trained convolutional neural networks for music audio tagging. In *Proc. of ISMIR LBD)*.

[53] Michael Pulis and Josef Bajada. 2021. Siamese Neural Networks for Content-based Cold-Start Music Recommendation.. In *Proc. of ACM RecSys* (Amsterdam, Netherlands). 719–723.

[54] Kiran R, Pradeep Kumar, and Bharat Bhasker. 2020. DNNRec: A novel deep learning based hybrid recommender system. *Expert Syst. Appl.* 144, C (2020), 14 pages.

[55] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning Transferable Visual Models From Natural Language Supervision. In *International Conference on Machine Learning*. https://api.semanticscholar.org/CorpusID:231591445

[56] Ramin Raziperchikolaei, Guannan Liang, and Young joo Chung. 2021. Shared Neural Item Representations for Completely Cold Start Problem. *Proc. of ACM RecSys* (2021).

[57] Scott Reed, Zeynep Akata, Honglak Lee, and Bernt Schiele. 2016. Learning deep representations of fine-grained visual descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 49–58.

[58] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proc. of EMNLP*. 3973–3983.

[59] Gang Ren, Lei Diao, Fanjia Guo, and Taeho Hong. 2024. A co-attention based multi-modal fusion network for review helpfulness prediction. *Information Processing & Management* 61, 1 (2024), 103573. doi:10.1016/j.ipm.2023.103573

[60] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Proc. of UAI* (Montreal, Quebec, Canada). 452–461.

[61] Francesco Ricci, Lior Rokach, and Bracha Shapira (Eds.). 2022. *Recommender Systems Handbook*. Springer US, New York, NY.

[62] Muhammad Saad Saeed, Muhammad Haris Khan, Shah Nawaz, Muhammad Haroon Yousaf, and Alessio Del Bue. 2022. Fusion and Orthogonal Projection for Improved Face-Voice Association. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 7057–7061.

[63] Muhammad Saad Saeed, Shah Nawaz, Muhammad Haris Khan, Muhammad Zaigham Zaheer, Karthik Nandakumar, Muhammad Haroon Yousaf, and Arif Mahmood. 2023. Single-branch Network for Multimodal Training. In *Proc. of ICASSP* (Rhodes Island, Greece). 1–5.

[64] Muhammad Saad Saeed, Shah Nawaz, Muhammad Zaigham Zaheer, Muhammad Haris Khan, Karthik Nandakumar, Muhammad Haroon Yousaf, Hassan Sajjad, Tom De Schepper, and Markus Schedl. 2024. Modality Invariant Multimodal Learning to Handle Missing Modalities: A Single-Branch Approach. arXiv:2408.07445 [cs.CV] https://arxiv.org/abs/2408.07445

[65] Scott Sanner, Krisztian Balog, Filip Radlinski, Ben Wedin, and Lucas Dixon. 2023. Large Language Models are Competitive Near Cold-start Recommenders for Language- and Item-based Preferences. In *Proc. of ACM RecSys* (Singapore, Singapore). 890–896.

[66] Markus Schedl, Hamed Zamani, Ching-Wei Chen, Yashar Deldjoo, and Mehdi Elahi. 2018. Current Challenges and Visions in Music Recommender Systems Research. *International Journal of Multimedia Information Retrieval* 7, 2 (2018), 95–116.

[67] Walid Shalaby, Sejoon Oh, Amir Afsharinejad, Srijan Kumar, and Xiquan Cui. 2022. M2TRec: Metadata-aware Multi-task Transformer for Large-scale and Cold-start free Session-based Recommendations. In *Proc. of ACM RecSys* (Seattle, WA, USA) *(RecSys '22)*. 573–578.

[68] Hao Tan and Mohit Bansal. 2019. LXMERT: Learning Cross-Modality Encoder Representations from Transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*.

[69] Michael Tschannen, Basil Mustafa, and Neil Houlsby. 2023. Clippo: Image-and-language understanding from pixels only. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 11006–11017.

[70] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2019. Representation Learning with Contrastive Predictive Coding. arXiv:1807.03748 [cs.LG]

[71] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Proc. of NeurIPS* (Long Beach, CA, US).

[72] Maksims Volkovs, Guangwei Yu, and Tomi Poutanen. 2017. DropoutNet: Addressing Cold Start in Recommender Systems. In *Proc. of NeurIPS*, Vol. 30.

[73] Ellen M. Voorhees. 1999. The TREC-8 Question Answering Track Report. In *The Eighth Text REtrieval Conference (TREC 8) (NIST Special Publication, Vol. 500-246)*, Ellen M. Voorhees and Dawn M. K. Harman (Eds.). National Institute of Standards and Technology (NIST), Gaithersburg, MD, USA, 77–82. https://trec.nist.gov/pubs/trec8/t8_proceedings.html

[74] Cheng Wang, Mathias Niepert, and Hui Li. 2018. LRMM: Learning to Recommend with Missing Modalities. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (Eds.). Association for Computational Linguistics, Brussels, Belgium, 3360–3370. doi:10.18653/v1/D18-1373

[75] Hu Wang, Yuanhong Chen, Congbo Ma, Jodie Avery, Louise Hull, and Gustavo Carneiro. 2023. Multi-Modal Learning With Missing Modality via Shared-Specific Feature Modelling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 15878–15887.

[76] Wenjie Wang, Xinyu Lin, Liuhui Wang, Fuli Feng, Yinwei Wei, and Tat-Seng Chua. 2023. Equivariant Learning for Out-of-Distribution Cold-start Recommendation. In *Proc. of ACM Multimedia* (Ottawa, Ontario, Canada). 903–914.

[77] Yinwei Wei, Xiang Wang, Qi Li, Liqiang Nie, Yan Li, Xuanping Li, and Tat-Seng Chua. 2021. Contrastive Learning for Cold-Start Recommendation. In *Proc. of ACM Multimedia* (Virtual Event, China). 5382–5390.

[78] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proc. of EMNLP System Demos* (Virtual Event). 38–45.

[79] Hanrui Wu, Chung Wang Wong, Jia Zhang, Yuguang Yan, Dahai Yu, Jinyi Long, and Michael K. Ng. 2023. Cold-Start Next-Item Recommendation by User-Item Matching and Auto-Encoders. *IEEE Transactions on Services Computing* 16, 4 (2023), 2477–2489.

[80] Peng Xu, Xiatian Zhu, and David A Clifton. 2023. Multimodal learning with transformers: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2023).

[81] Hong-Jian Xue, Xin-Yu Dai, Jianbing Zhang, Shujian Huang, and Jiajun Chen. 2017. Deep matrix factorization models for recommender systems. In *Proc. of IJCAI* (Melbourne, Australia). 3203–3209.

[82] Zhen Yang, Ming Ding, Tinglin Huang, Yukuo Cen, Junshuai Song, Bin Xu, Yuxiao Dong, and Jie Tang. 2024. Does Negative Sampling Matter? a Review With Insights Into its Theory and Applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 46, 8 (2024), 5692–5711. doi:10.1109/TPAMI.2024.3371473

[83] Chaohe Zhang, Xu Chu, Liantao Ma, Yinghao Zhu, Yasha Wang, Jiangtao Wang, and Junfeng Zhao. 2022. M3care: Learning with missing modalities in multimodal healthcare data. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2418–2428.

[84] Ziwei Zhu, Shahin Sefati, Parsa Saadatpanah, and James Caverlee. 2020. Recommendation for New Users and New Items via Randomized Training and Mixture-of-Experts Transformation. In *Proc. of ACM SIGIR* (Virtual Event, China). 1121–1130.

A Appendix

(a) SiBraR$_{best}$ (b) MuBraR$_{best}$
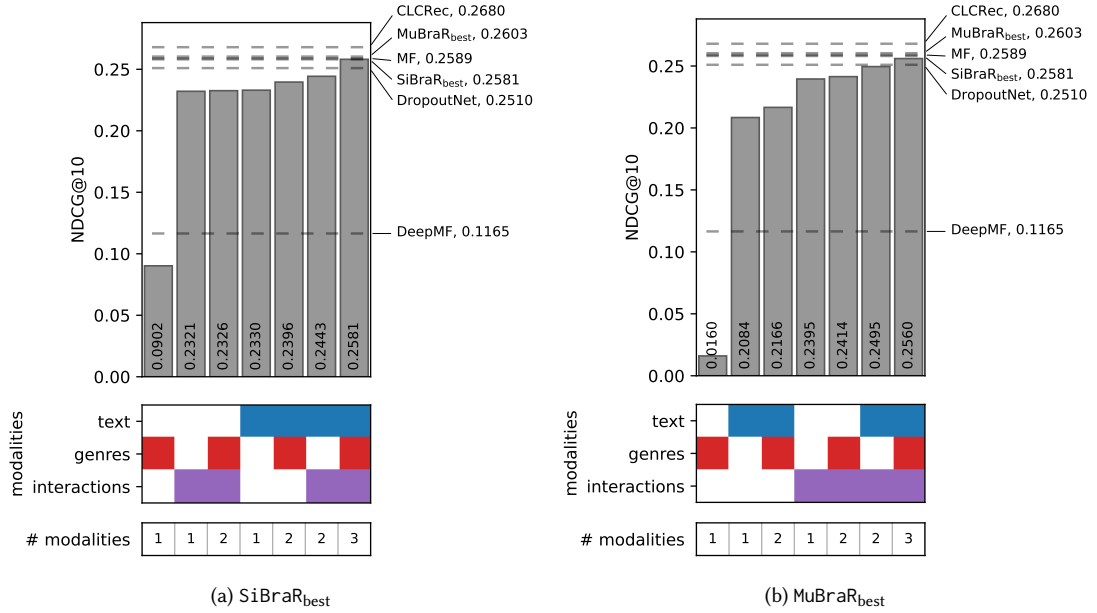
Fig. 14. NDCG@10 results of (a) SiBraR$_{best}$ and (b) MuBraR$_{best}$ trained on warm-start ML-1M with interactions, genres, and text modalities. The bars correspond to evaluations on the test set with varying evaluation modalities. If a modality is used, its block is filled with the corresponding color in the central plot. The bottom plot counts the number of modalities per bar. The gray dashed horizontal lines in the bar plot show the NDCG@10 of the best CFs and HRSs in warm-start ML-1M.
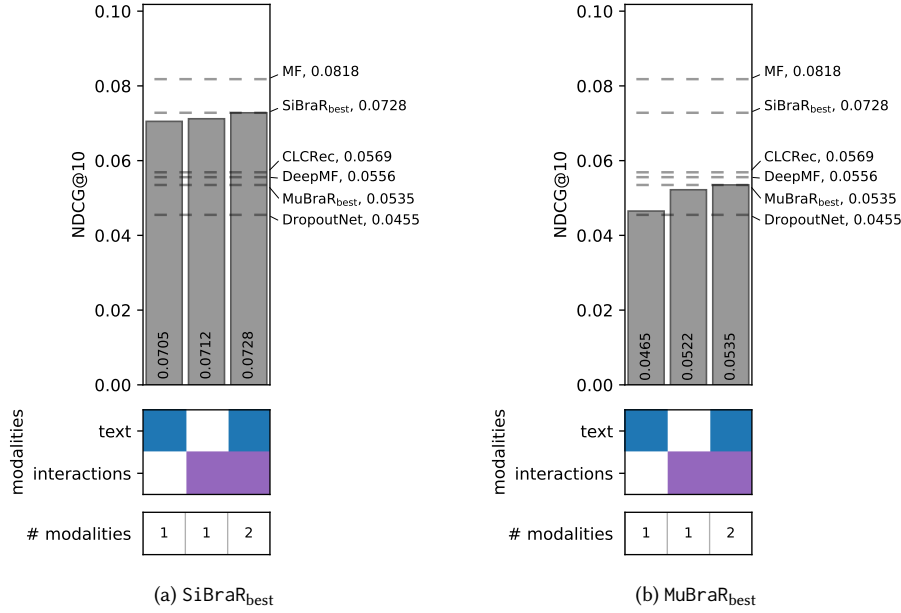
Fig. 15. NDCG@10 results of (a) SiBraR_best and (b) MuBraR_best trained on warm-start Amazon with interactions, and text modalities. The bars correspond to evaluations on the test set with varying evaluation modalities. If a modality is used, its block is filled with the corresponding color in the central plot. The bottom plot counts the number of modalities per bar. The gray dashed horizontal lines in the bar plot show the NDCG@10 of the best CFs and HRSs in warm-start Amazon.
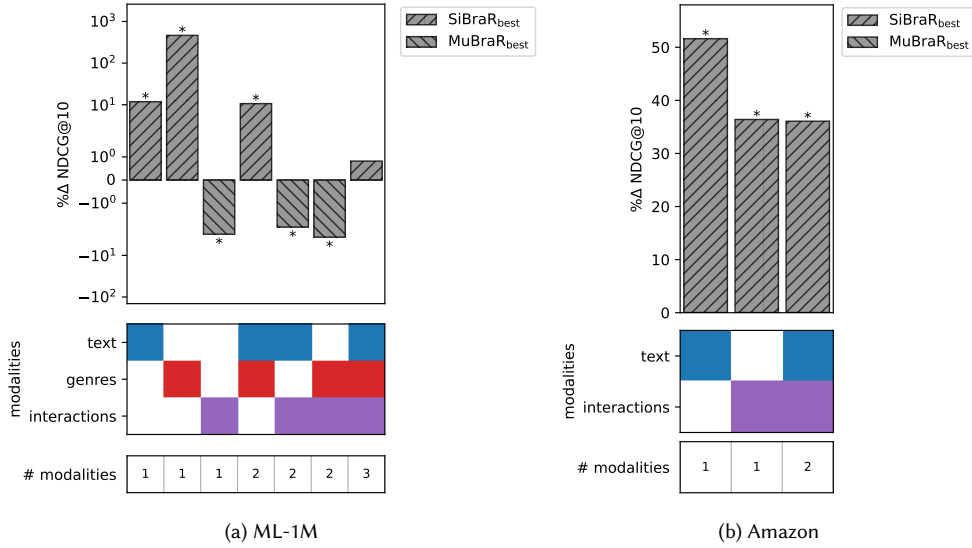
Fig. 16. NDCG@10 percent difference between SiBraR_best and MuBraR_best. For (a), both models are trained on warm-start ML-1M with interactions, genres, and text as modalities. For (b), they are trained on warm-start Amazon with interactions and text. Positive values indicate that SiBraR_best performs better than MuBraR_best on a specific set of modalities, while negative values indicate a poorer performance.

(a) MuBraR$_{SC}$ vs. MuBraR   (b) SiBraR$_{SC}$ vs. SiBraR$_{S}$
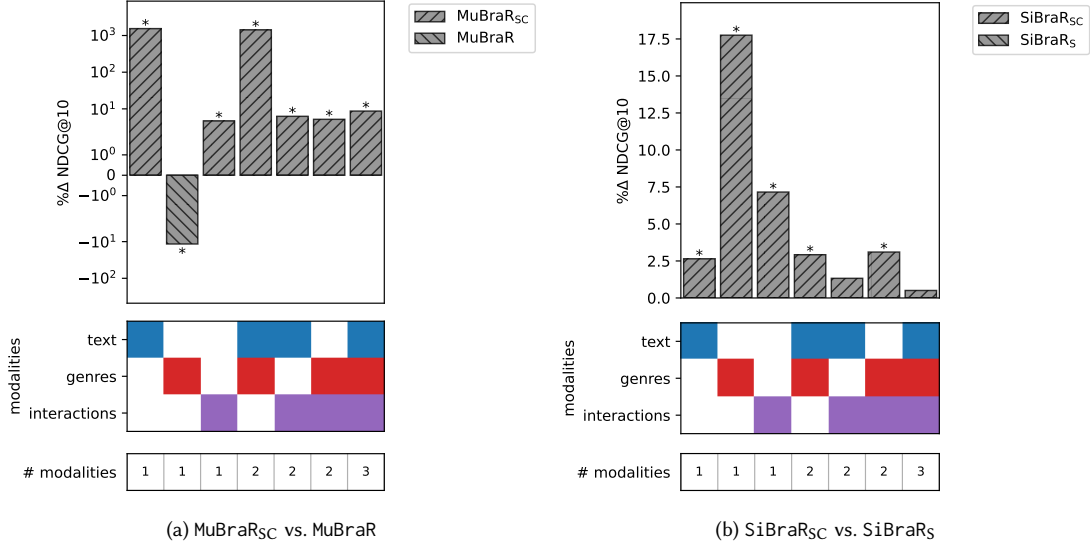
Fig. 17. NDCG@10 comparison for single- and multi-branch architectures when trained on all modalities of warm-start ML-1M, with and without utilizing modality sampling (S) and a contrastive loss (C). (a) shows the difference between MuBraR and MuBraR$_{SC}$, and (b) the difference between SiBraR$_{S}$ and SiBraR$_{SC}$. The colored squares in the middle plot indicate the modalities used during evaluation. Moreover, the bar plot shows the percent difference of NDCG@10, where positive values indicate that a contrastive loss is beneficial for recommendation quality, and worse if negative.

(a) MuBraR$_{SC}$ vs. MuBraR   (b) SiBraR$_{SC}$ vs. SiBraR$_{S}$
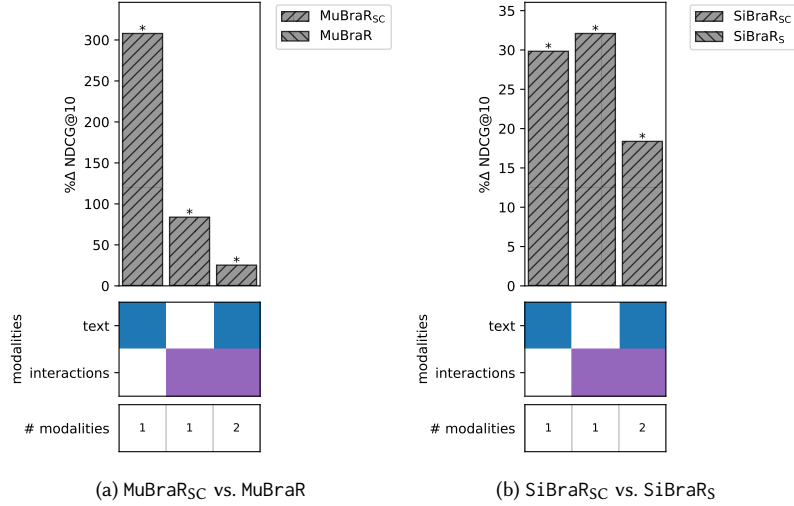
Fig. 18. NDCG@10 comparison for single- and multi-branch architectures when trained on all modalities of warm-start Amazon, with and without utilizing modality sampling (S) and a contrastive loss (C). (a) shows the difference between MuBraR and MuBraR$_{SC}$, and (b) the difference between SiBraR$_{S}$ and SiBraR$_{SC}$. The colored squares in the middle plot indicate the modalities used during evaluation. Moreover, the bar plot shows the percent difference of NDCG@10, where positive values indicate that a contrastive loss is beneficial for recommendation quality, and worse if negative.
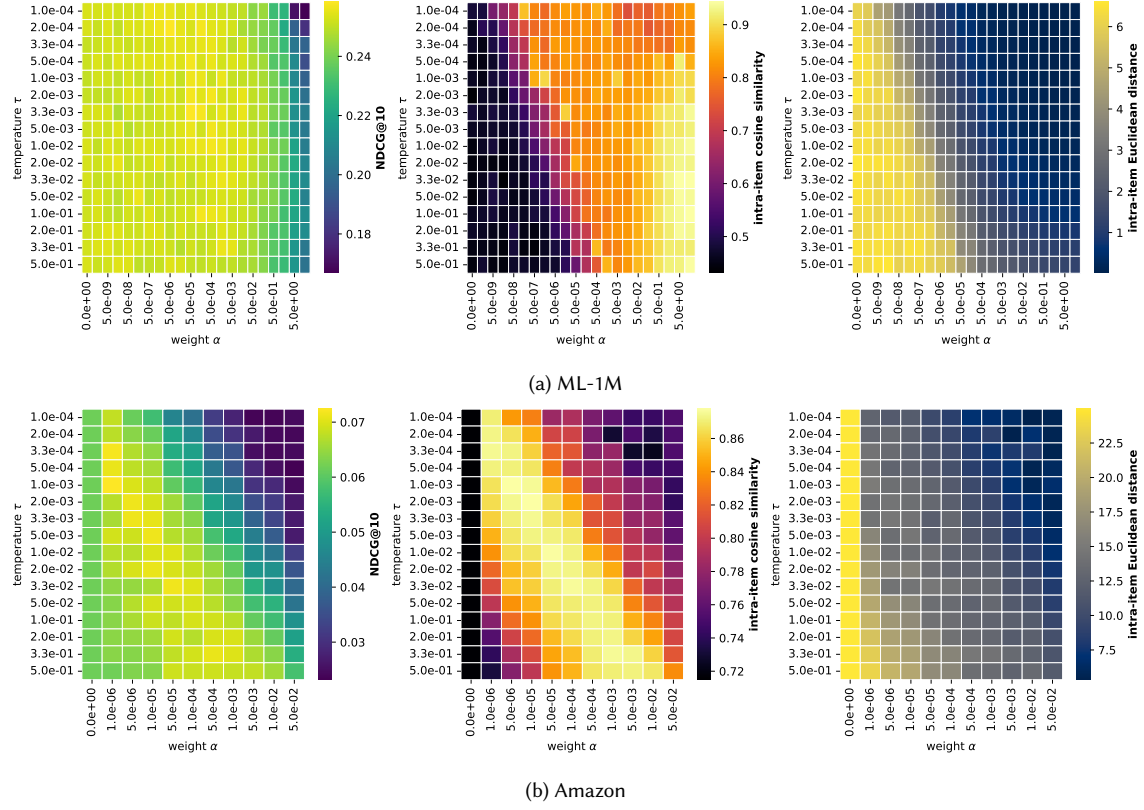
(a) ML-1M

(b) Amazon

Fig. 19. NDCG@10, intra-item cosine similarity and Euclidean distance results of SiBraR$_{best}$ on different combinations of contrastive loss temperature τ and weight α on warm-start (a) ML-1M and (b) Amazon, while all other hyperparameters per dataset are fixed. The left plots show the NDCG@10 results at test time, and the center and right plots show the intra-item cosine similarity and Euclidean distance, respectively. When contrastive loss $\alpha = 0$, i.e., when the loss does not affect training, the minor performance differences are due to randomness in the training pipeline.

	parameter group	parameter name	type	min	max	choices
MF		embedding size	int	32	512	
DeepMF	hidden item interaction layers	number of layers	int	0	1	
		layer size	int	16	512	
	hidden user interaction layers	number of layers	int	0	1	
		layer size	int	16	512	
		apply ReLU activation on interactions	bool			False, True
		apply ReLU activation on embeddings	bool			False, True
		embedding size	int	8	256	
		miminum recommendation score	float	1,00E-07	1,00E-02	
CLCRec	hidden side-information layers	number of layers	int	0	2	
		layer size	int	16	256	
	contrastive loss	weight	float	1,00E-06	1,00E-03	
		temperature	float	1,00E-03	1,00E-01	
		embedding size	int	8	512	
DropoutNet	hidden item interaction layers	number of layers	int	0	2	
		layer size	int	16	512	
	hidden item layers per side-information	number of layers	int	0	1	
		layer size	int	64	512	
	hidden item shared layers	number of layers	int	0	2	
		layer size	int	32	512	
	hidden user interaction layers	number of layers	int	0	2	
		layer size	int	16	512	
	hidden user layers per side-information	number of layers	int	0	1	
		layer size	int	8	24	
	hidden user shared layers	number of layers	int	0	2	
		layer size	int	32	512	
		embedding size	int	16	768	
		dropout rate	float	0.0	1.0	
MuBraR	hidden layers per modality	number of layers	int	1	3	
		layer size	int	128	512	
	contrastive loss	weight	float	0	1,00E-03	
		temperature	float	1,00E-03	5,00E-01	
		enable modality sampling	bool			False, True
		embedding size	int	128	768	
SiBraR	single-branch	number of hidden layers	int	0	5	
		hidden layer size	int	32	512	
		input size	int	16	768	
		apply batch normalization	bool			False, True
		apply ReLU output activation	bool			False, True
		normalize input	bool			False, True
	contrastive loss	weight	float	0	1,00E+00	
		temperature	float	1,00E-04	1,00E+00	
		embedding size	int	128	768	
	optimizer	learning rate	float	1,00E-05	5,00E-02	
		weight decay	float	0	5,00E-03	

Table 10. Summary of hyperparameters evaluated for various algorithms in this study. The second and third column refer to the parameter optimized, while the fourth which type of parameter it is. Lastly, the final three columns, *min*, *max*, and *choice*, indicate which hyperparameter ranges were considered. We use Sweeps module of Weights & Biases to explore the extensive parameter space, considering various distributions, e. g., *int_uniform* and *log_uniform* for integer parameters and float parameters spanning multiple orders of magnitude, respectively. Lastly, the group *optimizer* refers to the parameters of the Adam optimizer, which have been equally tuned for all models. The best hyperparameters were selected based on NDCG@10 performance on the validation set. When optimal parameters were found at the search space boundary, the space was expanded to ensure true optimality. Additionally, insights gained from hyperparameter analysis of one algorithm informed the initialization of parameter searches for other algorithms.

		Warm-Start			Item Cold-Start			User Cold-Start	
		ML-1M	Onion	Amazon	ML-1M	Onion	Amazon	ML-1M	Onion
BRS	Rand	0.0027	0.0008	0.0029	0.0309	0.0068	0.0231	0.0027	0.0009
	Pop	0.0798	0.0170	0.0515	0.0032	0.0064	0.0149	0.0508	0.0128
CF	MF	0.1806	0.1438	**0.1394**	0.0152	0.0056	0.0232	0.0520	0.0116
	DeepMF	0.0774	0.1224	0.1036	0.0032	0.0064	0.0149	0.0510	0.0079
HRS	CLCRec	**0.1851**	0.1363	0.1051	0.1153	0.1169	0.2199	0.0527	0.0128
	$\text{DropoutNet}_{\text{one}}$	0.1744	0.1052	0.0802	0.1488	0.1207	0.1906	0.0510	0.0137
	$\text{MuBraR}_{\text{one}}$	0.1813	0.1494	0.0977	0.1616	0.1944	0.2159	0.0504	0.0114
	$\text{SiBraR}_{\text{one}}$	0.1836	0.1495	0.1328	0.1736	0.1877	**0.2466**†	**0.0529**	0.0121
	$\text{DropoutNet}_{\text{best}}$	0.1744	0.1052	0.0887	0.1488	0.1618	0.1906	0.0510	**0.0155**
	$\text{MuBraR}_{\text{best}}$	0.1813	**0.1681**	0.0977	0.1775	0.2014	0.2159	0.0510	0.0121
	$\text{SiBraR}_{\text{best}}$	0.1820	0.1624	0.1328	**0.1869**†	**0.2114**†	**0.2466**†	0.0522	0.0155

Table 11. Recall@10 results for different BRSs, CFs, and HRSs. Each row indicates the results of the evaluated algorithm, and the columns blocks indicate the data scenario (warm start and user/item cold start) for the datasets (ML-1M, Onion, and Amazon). Bold indicates the best-performing RS, † indicates significant improvement over all others (paired t-tests with $p < 0.05$ considering Bonferroni correction).

		Warm-Start			Item Cold-Start			User Cold-Start	
		ML-1M	Onion	Amazon	ML-1M	Onion	Amazon	ML-1M	Onion
BRS	Rand	0.0042	0.0006	0.0003	0.0459	0.0061	0.0033	0.0435	0.0069
	Pop	0.1016	0.0136	0.0059	0.0076	0.0056	0.0021	0.4380	0.0927
CF	MF	0.2011	0.0818	**0.0158**	0.0292	0.0054	0.0035	0.4424	0.0858
	DeepMF	0.0989	0.0793	0.0118	0.0076	0.0056	0.0021	0.4253	0.0581
HRS	CLCRec	**0.2069**†	0.0764	0.0119	0.1644	0.0890	0.0314	0.4428	0.0923
	$\text{DropoutNet}_{\text{one}}$	0.1948	0.0663	0.0091	0.1832	0.0850	0.0276	0.4369	0.0912
	$\text{MuBraR}_{\text{one}}$	0.2013	0.0878	0.0111	0.2100	0.1386	0.0314	0.4328	0.0881
	$\text{SiBraR}_{\text{one}}$	0.1983	0.0905	0.0151	0.2108	0.1348	**0.0355**†	0.4497	0.0852
	$\text{DropoutNet}_{\text{best}}$	0.1948	0.0663	0.0100	0.1832	0.1109	0.0276	0.4369	0.1008
	$\text{MuBraR}_{\text{best}}$	0.2013	**0.1097**	0.0111	0.2300	0.1432	0.0314	0.4371	0.0873
	$\text{SiBraR}_{\text{best}}$	0.1977	0.1063	0.0151	**0.2350**†	**0.1449**	**0.0355**†	**0.4534**	**0.1044**

Table 12. Precision@10 results for different BRSs, CFs, and HRSs. Each row indicates the results of the evaluated algorithm, and the columns blocks indicate the data scenario (warm start and user/item cold start) for the datasets (ML-1M, Onion, and Amazon). Bold indicates the best-performing RS, † indicates significant improvement over all others (paired t-tests with $p < 0.05$ considering Bonferroni correction).

		Warm-Start			Item Cold-Start			User Cold-Start	
		ML-1M	Onion	Amazon	ML-1M	Onion	Amazon	ML-1M	Onion
BRS	Rand	0.0118	0.0017	0.0007	0.1105	0.0178	0.0103	0.1077	0.0231
	Pop	0.2581	0.0423	0.0189	0.0246	0.0133	0.0076	0.6993	0.2125
CF	MF	0.4552	0.2658	**0.0681**†	0.0702	0.0114	0.0114	0.6996	0.1978
	DeepMF	0.2305	0.2463	0.0438	0.0246	0.0133	0.0076	0.5698	0.1274
HRS	CLCRec	**0.4717**	0.2564	0.0448	0.2859	0.2973	0.1272	0.6950	0.2114
	DropoutNet$_{one}$	0.4489	0.2087	0.0317	0.3070	0.2147	0.0897	**0.7004**	0.2011
	MuBraR$_{one}$	0.4600	0.2642	0.0426	**0.5353**†	**0.4059**	0.1150	0.6976	0.2005
	SiBraR$_{one}$	0.4627	0.2724	0.0580	0.4635	0.3877	**0.1364**†	0.6627	0.1998
	DropoutNet$_{best}$	0.4489	0.2087	0.0348	0.3070	0.2969	0.0897	**0.7004**	0.2317
	MuBraR$_{best}$	0.4600	**0.3132**	0.0426	0.5220	0.3938	0.1150	0.6516	0.2127
	SiBraR$_{best}$	0.4627	0.3111	0.0580	0.5181	0.3885	**0.1364**†	0.6541	**0.2406**

Table 13. MRR@10 results for different BRSs, CFs, and HRSs. Each row indicates the results of the evaluated algorithm, and the columns blocks indicate the data scenario (warm start and user/item cold start) for the datasets (ML-1M, Onion, and Amazon). Bold indicates the best-performing RS, † indicates significant improvement over all others (paired t-tests with $p < 0.05$ considering Bonferroni correction).

		Warm-Start			Item Cold-Start			User Cold-Start	
		ML-1M	Onion	Amazon	ML-1M	Onion	Amazon	ML-1M	Onion
BRS	Rand	0.0007	0.0003	0.0006	0.0098	0.0019	0.0072	0.0010	0.0003
	Pop	0.0360	0.0056	0.0169	0.0011	0.0016	0.0055	0.0337	0.0056
CF	MF	0.0924	0.0739	**0.0618**†	0.0048	0.0011	0.0077	0.0332	0.0048
	DeepMF	0.0312	0.0615	0.0392	0.0011	0.0016	0.0055	0.0272	0.0026
HRS	CLCRec	**0.0975**	0.0726	0.0405	0.0498	0.0753	0.0974	0.0329	0.0055
	DropoutNet$_{one}$	0.0889	0.0499	0.0282	0.0616	0.0519	0.0675	0.0328	0.0058
	MuBraR$_{one}$	0.0933	0.0734	0.0384	0.0965	**0.1181**	0.0877	0.0335	0.0050
	SiBraR$_{one}$	0.0948	0.0747	0.0523	0.0898	0.1100	**0.1054**†	**0.0343**	0.0054
	DropoutNet$_{best}$	0.0889	0.0499	0.0309	0.0616	0.0777	0.0675	0.0328	0.0072
	MuBraR$_{best}$	0.0933	**0.0862**	0.0384	0.1098	0.1146	0.0877	0.0318	0.0056
	SiBraR$_{best}$	0.0942	0.0840	0.0523	**0.1134**†	0.1148	**0.1054**†	0.0330	**0.0073**

Table 14. MAP@10 results for different BRSs, CFs, and HRSs. Each row indicates the results of the evaluated algorithm, and the columns blocks indicate the data scenario (warm start and user/item cold start) for the datasets (ML-1M, Onion, and Amazon). Bold indicates the best-performing RS, † indicates significant improvement over all others (paired t-tests with $p < 0.05$ considering Bonferroni correction).

		Warm-Start			Item Cold-Start			User Cold-Start	
		ML-1M	Onion	Amazon	ML-1M	Onion	Amazon	ML-1M	Onion
BRS	Rand	0.0027	0.0005	0.0006	0.0299	0.0054	0.0056	0.0049	0.0014
	Pop	0.0726	0.0124	0.0104	0.0036	0.0048	0.0036	0.0833	0.0203
CF	MF	0.1564	0.0876	**0.0280**	0.0166	0.0045	0.0060	0.0851	0.0184
	DeepMF	0.0711	0.0799	0.0210	0.0036	0.0048	0.0036	0.0830	0.0126
HRS	CLCRec	**0.1610**†	0.0827	0.0212	0.1143	0.0834	0.0534	0.0860	0.0202
	DropoutNet$_{one}$	0.1514	0.0679	0.0161	0.1355	0.0814	0.0469	0.0835	0.0211
	MuBraR$_{one}$	0.1567	0.0934	0.0197	0.1517	0.1331	0.0531	0.0826	0.0183
	SiBraR$_{one}$	0.1572	0.0947	0.0268	0.1580	0.1291	**0.0602**†	**0.0866**	0.0189
	DropoutNet$_{best}$	0.1514	0.0679	0.0178	0.1355	0.1076	0.0469	0.0835	0.0240
	MuBraR$_{best}$	0.1567	**0.1112**	0.0197	0.1678	0.1381	0.0531	0.0835	0.0195
	SiBraR$_{best}$	0.1561	0.1076	0.0268	**0.1743**†	**0.1413**	**0.0602**†	0.0857	**0.0241**

Table 15. F-score@10 results for different BRSs, CFs, and HRSs. Each row indicates the results of the evaluated algorithm, and the columns blocks indicate the data scenario (warm start and user/item cold start) for the datasets (ML-1M, Onion, and Amazon). Bold indicates the best-performing RS, † indicates significant improvement over all others (paired t-tests with $p < 0.05$ considering Bonferroni correction).

		Warm-Start			Item Cold-Start			User Cold-Start	
		ML-1M	Onion	Amazon	ML-1M	Onion	Amazon	ML-1M	Onion
BRS	Rand	**1.0000**	**0.9799**	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**0.8484**	**0.3853**
	Pop	0.0288	0.0024	0.0041	0.0303	0.0073	0.0239	0.0032	0.0009
CF	MF	0.5156	0.4712	0.7575	0.0303	0.0073	0.0239	0.0032	0.0009
	DeepMF	0.3741	0.3705	0.8875	0.0303	0.0073	0.0239	0.0032	0.0009
HRS	CLCRec	0.4765	0.4417	0.8667	0.6667	0.9993	**1.0000**	0.0055	0.0010
	DropoutNet$_{one}$	0.4668	0.4799	0.4044	0.8818	0.8589	0.9737	0.0032	0.0028
	MuBraR$_{one}$	0.4862	0.5370	0.8870	0.7242	0.9280	**1.0000**	0.0032	0.0009
	SiBraR$_{one}$	0.5008	0.6445	0.7204	0.9606	0.9309	0.9928	0.0055	0.0186
	DropoutNet$_{best}$	0.4668	0.4799	0.4989	0.8818	0.8905	0.9737	0.0032	0.0055
	MuBraR$_{best}$	0.4862	0.4707	0.8870	0.8061	0.9148	**1.0000**	0.0032	0.0010
	SiBraR$_{best}$	0.4568	0.5855	0.7204	0.7636	0.9309	0.9928	0.0100	0.0180

Table 16. Coverage@10 results for different BRSs, CFs, and HRSs. Each row indicates the results of the evaluated algorithm, and the columns blocks indicate the data scenario (warm start and user/item cold start) for the datasets (ML-1M, Onion, and Amazon). Bold indicates the best-performing RS.

	Warm-Start			Item Cold-Start			User Cold-Start	
	ML-1M	Onion	Amazon	ML-1M	Onion	Amazon	ML-1M	Onion
MuBraR	**0.1823**	0.1237	0.0875	0.1431	0.1727	0.2062	0.0510	0.0113
MuBraR$_S$	0.1641	0.1523	0.0918	0.1583	0.2031	0.1920	0.0498	0.0129
MuBraR$_{SC}$	0.1813	**0.1681**	0.0977	0.1775	0.2014	0.2159	0.0510	0.0121
SiBraR$_S$	0.1798	0.1479	0.1159	0.1848	0.2024	0.1573	0.0502	**0.0155**
SiBraR$_{SC}$	0.1820	0.1624	**0.1328**†	**0.1869**	**0.2114**†	**0.2466**†	**0.0522**	0.0146

Table 17. Recall@10 results for the best-performing SiBraR and MuBraR variants on different warm- and user and item cold-start datasets (ML-1M, Onion, and Amazon). Bold indicates the best-performing RSs, † indicates significant improvement over all other variants (paired t-tests with $p < 0.05$ and Bonferroni correction).

	Warm-Start			Item Cold-Start			User Cold-Start	
	ML-1M	Onion	Amazon	ML-1M	Onion	Amazon	ML-1M	Onion
MuBraR	**0.2014**	0.0750	0.0099	0.1952	0.1264	0.0295	0.4381	0.0846
MuBraR$_S$	0.1890	0.0982	0.0104	0.2107	0.1416	0.0277	0.4363	0.0846
MuBraR$_{SC}$	0.2013	**0.1097**†	0.0111	0.2300	0.1432	0.0314	0.4371	0.0873
SiBraR$_S$	0.1967	0.0979	0.0131	0.2342	0.1389	0.0230	0.4430	**0.1044**
SiBraR$_{SC}$	0.1977	0.1063	**0.0151**†	**0.2350**	**0.1449**	**0.0355**†	**0.4534**	0.1025

Table 18. Precision@10 results for the best-performing SiBraR and MuBraR variants on different warm- and user and item cold-start datasets (ML-1M, Onion, and Amazon). Bold indicates the best-performing RSs, † indicates significant improvement over all other variants (paired t-tests with $p < 0.05$ and Bonferroni correction).

	Warm-Start			Item Cold-Start			User Cold-Start	
	ML-1M	Onion	Amazon	ML-1M	Onion	Amazon	ML-1M	Onion
MuBraR	0.4627	0.2360	0.0354	0.4171	0.3724	0.1123	**0.6991**	0.1866
MuBraR$_S$	0.4408	0.2908	0.0404	0.4734	0.3903	0.0961	0.6902	0.2003
MuBraR$_{SC}$	0.4600	**0.3132**	0.0426	0.5220	**0.3938**	0.1150	0.6516	0.2127
SiBraR$_S$	0.4592	0.2797	0.0479	**0.5505**†	0.3733	0.0849	0.6953	**0.2406**
SiBraR$_{SC}$	**0.4627**	0.3111	**0.0580**†	0.5181	0.3885	**0.1364**†	0.6541	0.2283

Table 19. MRR@10 results for the best-performing SiBraR and MuBraR variants on different warm- and user and item cold-start datasets (ML-1M, Onion, and Amazon). Bold indicates the best-performing RSs, † indicates significant improvement over all other variants (paired t-tests with $p < 0.05$ and Bonferroni correction).

	Warm-Start			Item Cold-Start			User Cold-Start	
	ML-1M	Onion	Amazon	ML-1M	Onion	Amazon	ML-1M	Onion
MuBraR	0.0937	0.0617	0.0319	0.0779	0.1008	0.0868	**0.0333**	0.0046
MuBraR$_S$	0.0844	0.0761	0.0369	0.0898	0.1132	0.0728	0.0322	0.0053
MuBraR$_{SC}$	0.0933	**0.0862**	0.0384	0.1098	0.1146	0.0877	0.0318	0.0056
SiBraR$_S$	0.0928	0.0738	0.0432	**0.1140**	0.1093	0.0639	0.0328	**0.0073**
SiBraR$_{SC}$	**0.0942**	0.0840	**0.0523**†	0.1134	**0.1148**	**0.1054**†	0.0330	0.0068

Table 20. MAP@10 results for the best-performing SiBraR and MuBraR variants on different warm- and user and item cold-start datasets (ML-1M, Onion, and Amazon). Bold indicates the best-performing RSs, † indicates significant improvement over all other variants (paired t-tests with $p < 0.05$ and Bonferroni correction).

	Warm-Start			Item Cold-Start			User Cold-Start	
	ML-1M	Onion	Amazon	ML-1M	Onion	Amazon	ML-1M	Onion
MuBraR	**0.1575**	0.0782	0.0176	0.1385	0.1202	0.0502	0.0836	0.0183
MuBraR$_S$	0.1446	0.0999	0.0185	0.1512	0.1375	0.0470	0.0819	0.0197
MuBraR$_{SC}$	0.1567	**0.1112**†	0.0197	0.1678	0.1381	0.0531	0.0835	0.0195
SiBraR$_S$	0.1541	0.0982	0.0233	0.1724	0.1359	0.0388	0.0826	**0.0241**
SiBraR$_{SC}$	0.1561	0.1076	**0.0268**†	**0.1743**	**0.1413**	**0.0602**†	**0.0857**	0.0234

Table 21. F-score@10 results for the best-performing SiBraR and MuBraR variants on different warm- and user and item cold-start datasets (ML-1M, Onion, and Amazon). Bold indicates the best-performing RSs, † indicates significant improvement over all other variants (paired t-tests with $p < 0.05$ and Bonferroni correction).

	Warm-Start			Item Cold-Start			User Cold-Start	
	ML-1M	Onion	Amazon	ML-1M	Onion	Amazon	ML-1M	Onion
MuBraR	0.5095	**0.7009**	**0.9299**	0.9424	**0.9890**	**1.0000**	0.0032	0.0009
MuBraR$_S$	0.4274	0.4967	0.8832	**0.9758**	0.9530	**1.0000**	0.0032	0.0023
MuBraR$_{SC}$	0.4862	0.4707	0.8870	0.8061	0.9148	**1.0000**	0.0032	0.0010
SiBraR$_S$	**0.5362**	0.4957	0.6474	0.8182	0.9552	**1.0000**	**0.0119**	**0.0180**
SiBraR$_{SC}$	0.4568	0.5855	0.7204	0.7636	0.9309	0.9928	0.0100	0.0087

Table 22. Coverage@10 results for the best-performing SiBraR and MuBraR variants on different warm- and user and item cold-start datasets (ML-1M, Onion, and Amazon). Bold indicates the best-performing RSs.

	# item modalities		
	1	2	3
MuBraR	0.0866	0.1595	0.2355
MuBraR$_S$	0.1260	0.2060	0.2427
MuBraR$_{SC}$	0.1547	0.2358	0.2560
SiBraR$_S$	0.1734	0.2332	0.2568
SiBraR$_{SC}$	**0.1851**	**0.2388**	**0.2581**

Table 23. NDCG@10 results of different SiBraR and MuBraR variants trained on warm-start ML-1M with the same set of item modalities (interactions, genres, and text) and evaluated on subsets of those. NDCG@10 values of different subsets with the same number of modalities are averaged. Bold indicates the best-performing algorithm per the number of modalities used.

| | # item modalities | |
	1	2
MuBraR	0.0199	0.0427
MuBraR$_S$	0.0442	0.0508
MuBraR$_{SC}$	0.0494	0.0535
SiBraR$_S$	0.0541	0.0615
SiBraR$_{SC}$	**0.0708**	**0.0728**

Table 24. NDCG@10 results of different SiBraR and MuBraR variants trained on warm-start Amazon with the same set of item modalities (interactions, text) and evaluated on subsets of those. NDCG@10 values of different subsets with the same number of modalities are averaged. Bold indicates the best-performing algorithm per the number of modalities used.

| | # item modalities | |
	1	2
MuBraR	0.1502	0.2511
MuBraR$_S$	0.1748	0.2620
MuBraR$_{SC}$	0.1982	0.2949
SiBraR$_S$	0.2374	**0.3046**
SiBraR$_{SC}$	**0.2413**	0.2994

Table 25. NDCG@10 results of different SiBraR and MuBraR variants trained on cold-start ML-1M with the same set of item modalities (interactions, genres, and text) and evaluated on subsets of those (excluding interactions). NDCG@10 values of different subsets with the same number of modalities are averaged. Bold indicates the best-performing algorithm per the number of modalities used.
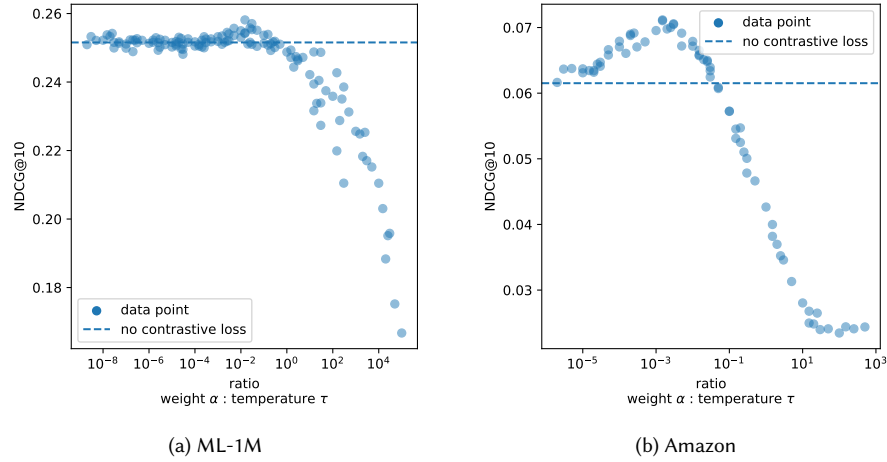

(a) ML-1M

(b) Amazon

Fig. 20. NDCG@10 results of SiBraR$_{best}$ on different combinations of contrastive loss temperature τ and weight α on warm-start (a) ML-1M and (b) Amazon, while all other hyperparameters per dataset are fixed. Each point indicates the performance of a run with a certain ratio α/τ and the resulting NDCG@10. The horizontal dashed line shows the average performance for runs where the contrastive loss has no effect on training, i. e., when $\alpha = 0$.

MuBraR MuBraR$_S$ MuBraR$_{SC}$ SiBraR$_S$ SiBraR$_{SC}$

● text ● genres ● interactions

(a) ML-1M

MuBraR MuBraR$_S$ MuBraR$_{SC}$ SiBraR$_S$ SiBraR$_{SC}$
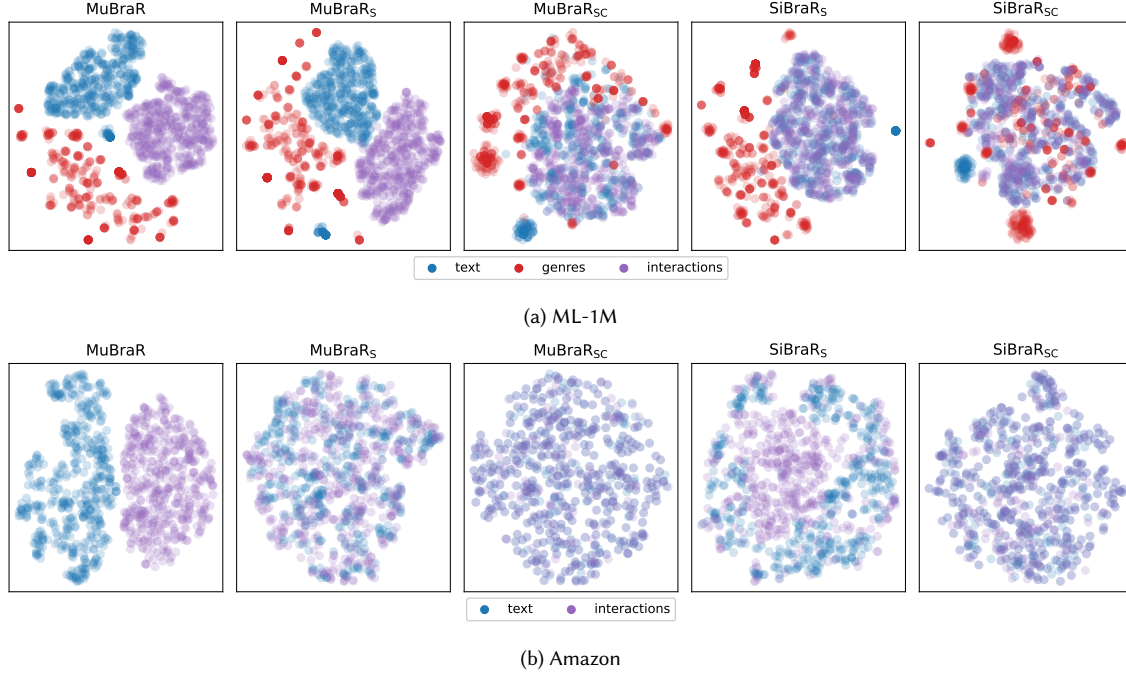

● text ● interactions

(b) Amazon

Fig. 21. T-SNE projected embeddings of the modality-specific embeddings for all MuBraR and SiBraR variants on warm-start ML-1M (a) and Amazon (b). While the different modalities form their own clusters in the embedding space of MuBraR models (1st), for models using a contrastive loss (MuBraR$_{SC}$, 3rd and SiBraR$_{SC}$, 5th), the modalities end up in similar regions in the projection space. Genres hardly overlap with other modalities in ML-1M, which we attribute to the low number of genres and thus similar embeddings for all items.

● genres
● learned embedding

● image
● learned embedding

● text
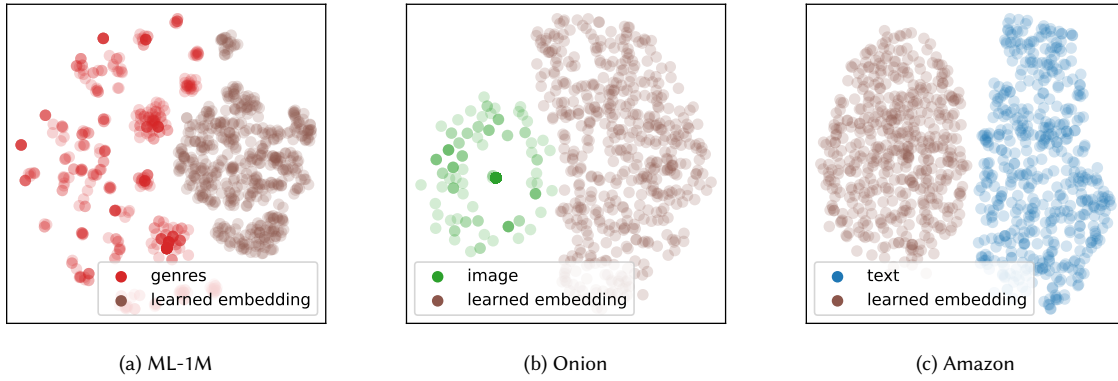● learned embedding

(a) ML-1M (b) Onion (c) Amazon

Fig. 22. T-SNE projected embeddings of the modality-specific embeddings best CLCRec algorithms on warm-start ML-1M (a), Onion (b) and Amazon (c). All modalities form their own clusters in the embedding space across datasets, indicating low proximity of different modalities.

	ML-1M		Onion		Amazon	
	intra	inter	intra	inter	intra	inter
Euclidean distance	2.569	1.811	7.306	4.890	6.821	7.403
cosine similarity	0.044	0.153	0.000	0.051	0.432	0.162

Table 26. Average inter- and intra-item Euclidean distance (ED) and cosine similarity (CS) metrics for the best CLCRec models trained each warm-start dataset. While modalities of the same type are more similar in the embedding space than between modalities of the same item for CLCRec on the ML-1M and Onion datasets (lower ED and higher CS), the modalities of individual item are more similar than to the same modalities of other items for the Amazon dataset.

	ML-1M		Onion		Amazon	
	intra	inter	intra	inter	intra	inter
MuBraR	188.587	90.606	420.848	129.887	30.632	14.551
MuBraR$_S$	125.929	85.579	60.876	55.630	44.273	62.094
MuBraR$_{SC}$	18.158	18.193	13.814	15.683	**4.742**	**11.448**
SiBraR$_S$	2.244	2.463	9.347	10.218	24.956	34.775
SiBraR$_{SC}$	**0.756**	**1.191**	**4.161**	**5.950**	12.209	31.765

Table 27. Average inter- and intra-item Euclidean distance (ED) for SiBraR and MuBraR variants. For each warm-start dataset, all variants are trained and evaluated on the same set of modalities: (interactions, genres, text) for ML-1M, (interactions, genres, audio, image, text) for Onion, and (interactions, text) for Amazon. Bold indicates the algorithm where modality embeddings exhibit the closest average proximity within each group.

	ML-1M	Onion
MuBraR	1.0000	1.0000
MuBraR$_S$	1.0000	0.9999
MuBraR$_{SC}$	1.0000	0.9979
SiBraR$_S$	**0.9485**†	0.9008
SiBraR$_{SC}$	0.9928	**0.7424**†
Random	0.5000	0.2500

Table 28. Accuracy of random forest classifiers predicting the modality class of modality embeddings output by MuBraR and SiBraR variants trained on item cold-start Onion and ML-1M. We do not consider Amazon, as it works best in cold-start when trained on a single type of side-information, which makes the classification task trivial. Each result reports the average performance of 20 classifiers trained on different train–test splits and seeds. 80% of each cold-start test set is used for fitting the classifiers, and 20% for their evaluation. Bold indicates the variant with the least separable embedding space, † indicates significant worse classifier accuracies in comparison to all other methods (paired t-tests with $p < 0.05$ considering Bonferroni correction). The last row shows the expected accuracy for random guessing.